

# Satisfiability of General Intruder Constraints with and without a Set Constructor<sup>☆</sup>

Tigran Avanesov<sup>a</sup>, Yannick Chevalier<sup>b</sup>, Michaël Rusinowitch<sup>a</sup>, Mathieu Turuani<sup>a</sup>

<sup>a</sup> *Loria, Inria Nancy - Grand Est, Campus Scientifique — BP 239, 54506 Vandœuvre-lès-Nancy, France*

<sup>b</sup> *IRIT - Université Paul Sabatier, 118 route de Narbonne, 31020 Toulouse Cedex, France*

---

## Abstract

Many decision problems on security protocols can be reduced to solving so-called intruder constraints in Dolev Yao model. Most constraint solving procedures for protocol security rely on two properties of constraint systems called *monotonicity* and *variable-origination*. In this work we relax these restrictions by giving a decision procedure for solving general intruder constraints (that do not have these properties) that stays in NP. Our result extends a first work by L. Mazaré in several directions: we allow non-atomic keys, and an associative, commutative and idempotent symbol (for modeling sets). We also discuss several new applications of the results.

*Keywords:* ACI, deducibility constraints, Dolev-Yao deduction system, multiple intruders, security.

---

## 1. Introduction

Detecting flaws in security protocol specifications under the perfect cryptography assumption in Dolev-Yao intruder model is an approach that has

---

<sup>☆</sup>The work presented in this paper was partially supported by the FP7-ICT-2007-1 Project no. 216471, “AVANTSSAR: Automated Validation of Trust and Security of Service-oriented Architectures” (<http://www.avantssar.eu>) and FP7-ICT Project no. 256980, “NESSoS: Network of Excellence on Engineering Secure Future Internet Software Services and Systems” (<http://www.nessos-project.eu>).

*Email addresses:* [Tigran.Avanesov@loria.fr](mailto:Tigran.Avanesov@loria.fr) (Tigran Avanesov), [ychevali@irit.fr](mailto:ychevali@irit.fr) (Yannick Chevalier), [Michael.Rusinowitch@loria.fr](mailto:Michael.Rusinowitch@loria.fr) (Michaël Rusinowitch), [Mathieu.Turuani@loria.fr](mailto:Mathieu.Turuani@loria.fr) (Mathieu Turuani)

been extensively investigated in recent years [1, 2, 3, 4]. In particular symbolic constraint solving has proved to be a very successful approach in the area. It amounts to express the possibility of mounting an attack, e.g. the derivation of a secret, as a list of steps where for each step some message has to be derived from the current intruder knowledge. These steps correspond in general to the progression of the protocol execution, up to the last one which is the secret derivation.

Enriching standard Dolev-Yao intruder model with different equational theories [5, 6] like exclusive OR, modular exponentiation, Abelian groups, etc. [7, 8, 9] helps to find flaws that could not be detected considering free symbols only. A particularly useful theory is the theory of an *ACI* operator (that is associative commutative and idempotent) since it allows one to express sets in cryptographic protocols.

Up to one exception [10, 11], all proposed algorithms rely on two strong assumptions about the constraints to be processed: knowledge monotonicity and variable origination. Constraints satisfying this hypothesis are called *well-formed constraints* in the literature and they are not restrictive as these conditions hold when handling standard security problems with a single Dolev-Yao intruder. However, we will see that in some situations it can be quite useful to relax these hypotheses and consider *general constraints*, that is constraints without the restrictions above. General constraints naturally occur when considering security problems involving several non-communicating Dolev-Yao intruders (see § 2.1). Remark that if intruders can communicate during protocol execution, the model becomes attack-equivalent to one with a unique Dolev-Yao intruder [12].

### 1.1. Contributions of the paper

First, we will show that as for the standard case, in this more general framework it is still possible to derive an NP decision procedure for detecting attacks on a bounded number of protocol sessions (Sections 5, 4). Second, our result extends previous ones by allowing non-atomic keys and the usage of an associative commutative idempotent operator (Sections 3, 4) that can be used for instance to model sets of nodes in XML document (see § 2.2). Third, we will remark that the satisfiability procedure we obtain for *general constraints* is a non trivial extension of the one for *well-formed constraints* by showing that this procedure cannot be extended to handle operators with subterm convergent theories since satisfiability gets undecidable in this case (Appendix A). On the other hand it is known that satisfiability remains

decidable for the standard case of *well-formed constraints* with the same operator properties [13]. Finally we will sketch the potential applications of our results (Section 2).

### 1.2. Related works

The decision procedure for satisfiability of well-formed constraint systems can be used to decide the insecurity of cryptographic protocols with a bounded number of sessions [14]. In this domain, several works deviated from the perfect cryptography assumption and started to consider algebraic properties of functional symbols. For example properties of XOR operator and exponentiation were considered in [15, 8, 16, 17] and together with homomorphic symbol in [18]. Some algebraic properties (like associative and commutative symbol) make the insecurity problem undecidable [19].

All the works mentioned above consider systems of constraints with two restrictions namely knowledge monotonicity (the left-hand side of a constraint representing the current knowledge of the intruder is included into the left-hand side of the next one) and variable origination (variable appears first in the right-hand side of some constraint): this limitation is not impeding the solution of usual protocol insecurity problems since the constraints generated with an active Dolev-Yao intruder are of the required type. An attempt to swerve from well-formed constraints was made by Mazaré [10]. He considered “quasi well-formed” constraint systems by partially relaxing the knowledge monotonicity. Later, in his thesis [11], he raised a similar decidability problem, but now for general constraint systems. He succeeded to find a decision procedure for satisfiability of general constraint systems with the restriction that keys used for encryption are atomic. However to our knowledge no extension of Dolev-Yao deduction system to non-atomic key or to algebraic properties has been shown decidable for general constraint systems. Moreover, satisfiability of well-formed constraints with ACI theory was not considered before.

## 2. Motivating examples

### 2.1. Protocol analysis with several intruders

In the domain of security protocol analysis Dolev-Yao model is widely used in spite of its limitations. We propose here to consider instead of a powerful Dolev-Yao intruder that controls the whole network, several non

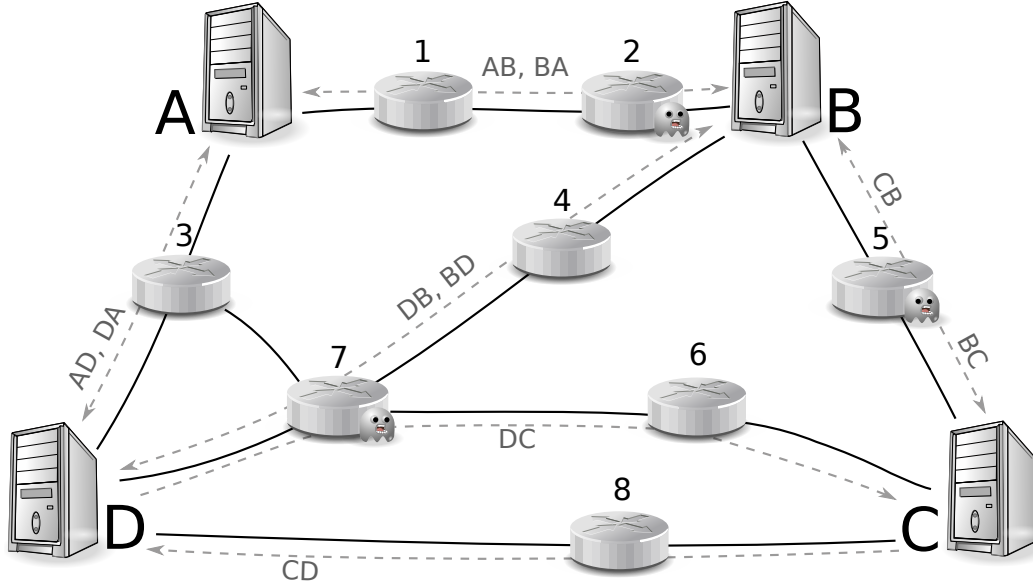


Figure 1: Untrusted routers

communicating Dolev-Yao intruders with smaller controlled domains. We give below an application of this model.

Suppose several agents ( $A, B, \dots$ , see Figure 1) execute a message exchange protocol (every agent has a finite list of actions in a send/receive format that is known to everybody). Due to their (long distance) layout they have to transmit data through routers (1, 2, 3...). The routing tables of all honest routers/agents are static (messages follow always the same path). Some routers (2, 5, 7) may be compromised: an intruder managed to install a device controlling input and output of the router or implanted there his malicious code. A message circulated via such an untrusted channel (e.g.  $DB$ ) is consumed by the corresponding compromised device (*local intruder*) (7) thereby increasing his knowledge. Moreover, a local intruder can forge and emit to an endpoint ( $C, B, D$ ) of any channel he controls ( $BD, DB, DC$ ) any message he can build using the content of his memory and some available transformations specified by a deduction system. Because of the network topology malicious routers have no means to communicate (there is no links between them, neither direct nor via other routers), but at some point the intruder can gather the knowledge of all the compromised routers (by physically collecting devices or reading their memory).

In this framework the *security problem* is to know whether it is possible to initially give instructions to compromised routers (e.g. by reprogramming malicious devices) to force such an execution that honest agents (that strictly follow their list of actions) will reveal some secret data to the intruder (i.e. intruder can build this data from the gathered at the end knowledge of all local intruders).

### 2.1.1. Formalizing the coordinated attack problem

To formalize the problem we introduce some notations and definitions that are more detailed in Subsection 3.1.

*Messages.* We consider first-order terms built from a set of function symbols (such as encryption, pairing, etc.), a set of constants  $\mathcal{A}$  (representing elementary pieces of data: texts, public keys, names of agents, etc. also called *atoms*) and a set of variables  $\mathcal{X}$ . Let  $\mathcal{T}$  be the set of all possible terms. For a term  $t$  we write  $\text{Vars}(t)$  the set of all variables in  $t$  (see Def. 3.9). A term  $t$  is a *ground term*, if  $\text{Vars}(t) = \emptyset$ . The set of ground terms is denoted by  $\mathcal{T}_g$ . We assume that terms are interpreted modulo an equational theory and that we can compute for every term  $t$  a unique normal form denoted by  $\lceil t \rceil$  modulo this equational theory. (We will focus later on the special case where we have a function symbol  $\cdot$  and the theory is generated by the commutativity, associativity and idempotency of  $\cdot$ ). A term  $t$  is *normalized* if  $t = \lceil t \rceil$ . Two terms  $p$  and  $q$  are *equivalent*, if  $\lceil p \rceil = \lceil q \rceil$ . Given a set of terms  $T$  we define  $\lceil T \rceil = \{\lceil t \rceil : t \in T\}$ . More details are given in Section 3.

We define a substitution  $\sigma = \{x_1 \mapsto t_1, \dots, x_k \mapsto t_k\}$  (where  $x_i \in \mathcal{X}$  and  $t_i \in \mathcal{T}$ ) to be the mapping  $\sigma : \mathcal{T} \rightarrow \mathcal{T}$ , such that  $t\sigma$  is a term obtained by replacing, for all  $i$ , each occurrence of variable  $x_i$  by the corresponding term  $t_i$ . The set of variables  $\{x_1, \dots, x_k\}$  is called the *domain* of  $\sigma$  and denoted by  $\text{dom}(\sigma)$ . If  $T \subseteq \mathcal{T}$ , then by definition  $T\sigma = \{t\sigma : t \in T\}$ . A substitution  $\sigma$  is *ground* if for any  $i \in \{1, \dots, k\}$ ,  $t_i$  is ground. We will say that the substitution  $\sigma$  is *normalized*, if for all  $x \in \text{dom}(\sigma)$ ,  $x\sigma$  is normalized.

*Agents.* We will call communicating parties *agents*. Every agent is identified by its name. We denote a set of agent names as  $A$ .

*Channels.* Any two agents  $a$  and  $b$  communicate through a channel denoted as  $a \rightarrow b$ . We will suppose, that channels are directed. The set of all channels is denoted as  $\mathbb{C}$ . A channel supports a queue of messages: for example, if  $a$  sends sequentially two messages to  $b$  (via channel  $a \rightarrow b$ ), then  $b$  cannot

process the second message before the first one; the sent messages are stored in queue to be processed in order of arrival.

*Agents behavior.* We define a protocol session  $PS = \{\langle a_i, l_i \rangle\}_{i=1, \dots, k}$  as a finite set of pairs of an agent name and a finite list of actions to be executed by this agent<sup>1</sup>. We also suppose that  $\text{Vars}(l_i) \cap \text{Vars}(l_j) = \emptyset$ , for all  $i \neq j$  (where  $\text{Vars}(\cdot)$  is naturally extended on lists of actions).

Every action is of receiving type  $?_f r$  or sending type  $!_t s$ , where

- $f$  is an agent name, whom a message is to be received from;
- $r$  is a term (a template for the message) expected to be received from  $f$ ;
- $t$  is an agent name, whom the message is expected to be sent to;
- $s$  is a term (a template for the message) to be sent to  $t$ .

Let us consider any agent  $a \in A$  participating in the protocol session  $PS$  and let  $\langle a, \{\rho_i\}_{i=1, \dots, k} \rangle \in PS$ .

Case 1. If  $\rho_1 = ?_{f_1} r_1$  then the first action agent  $a$  can do, is to accept a message  $m$ , admittedly from agent  $f_1$  on channel  $f_1 \rightarrow a$ , matching the pattern  $r_1$ , i.e. such that  $\lceil r_1 \sigma \rceil = \lceil m \rceil$  for some substitution  $\sigma$ . Agent is blocked (does not execute any other actions) by awaiting a message. If  $a$  receives a message that does not match the expected pattern, then  $a$  terminates his participation in  $PS$ . Note that no notification is sent to the sender, thus a sender continues his execution<sup>2</sup>. Once  $a$  has received message  $m$  matching the pattern  $r_1$  with substitution  $\sigma$ , he instantiate  $\text{Vars}(r_1)$  with  $\sigma$  and execute his remaining actions using these values, i.e.  $a$  moves to a state where the list of actions to be executed is  $\{\rho_i \sigma\}_{i=2, \dots, k}$ , with

$$\rho \sigma = \begin{cases} ?_f(r\sigma), & \text{if } \rho = ?_f r; \\ !_t(s\sigma), & \text{if } \rho = !_t s. \end{cases}$$

---

<sup>1</sup>For simplicity, we suppose that for a protocol session, one agent cannot have more than one list of actions to execute, but this restriction can be relaxed.

<sup>2</sup>A way to model another behavior, is to explicitly provide for every sending a succedent receive of an acknowledge message and for every receive a succedent send of an acknowledge message.

We will say that an action  $\rho$  is ground, if  $\rho = ?_f r$  and  $r$  is a ground term; or  $\rho = !_t s$  and  $s$  is ground.

Case 2. If  $\rho_1$  is  $!_{t_1} s_1$  then the first action of agent  $a$  is sending message  $s_1$  to agent  $t_1$  (i.e. putting it to channel  $a \rightarrow t_1$ ) and then, moving to a state where  $\{\rho_i\}_{i=2,\dots,k}$  has to be executed.

We suppose that agents cannot have a sending pattern that contains variables not instantiated before, i.e. for any  $\langle a, \rho_1 \cdots \rho_{k_a} \rangle \in PS$  if  $\rho_i = !_t s$  then for any variable  $x \in \text{Vars}(s)$  there exists  $j < i$  such that  $\rho_j = ?_f r$  and  $x \in \text{Vars}(r)$ .

*Intruder model.* We assume that some communication channels are controlled by  $N$  local intruders  $\{I_i\}_{i=1,\dots,N}$  and there is no channel controlled by more than one intruders. We introduce an *intruders layout* represented by a function  $\iota : \mathbb{C} \mapsto \mathbb{I} \cup \{\emptyset\}$  mapping every channel to the local intruder that controls it if there is one, to  $\emptyset$  otherwise.

Every intruder  $I$  is given some initial knowledge  $K_I^0$  that is a set of ground terms. Once an agent sends a message via a channel controlled by intruder, the intruder reads it and blocks it. Reading the message means extending intruder's current knowledge with this message. An intruder controlling a channel can generate a message from his knowledge using deduction rules and send it to its endpoint.

We now specify the intruder capabilities:

**Definition 2.1.** A *rule* is a tuple of terms written as  $s_1, \dots, s_k \rightarrow s$ , where  $s_1, \dots, s_k, s$  are terms. A *deduction system*  $\mathcal{D}$  is a set of rules.

From now to the end of this section rules are assumed to belong to a fixed deduction system  $\mathcal{D}$ .

**Definition 2.2.** A *ground instance* of a rule  $d = s_1, \dots, s_k \rightarrow s$  is a rule  $l = l_1, \dots, l_k \rightarrow r$  where  $l_1, \dots, l_k, r$  are ground terms and there exists  $\sigma$  — ground substitution, such that  $l_i = s_i \sigma$ , for  $i = 1, \dots, k$  and  $r = s \sigma$ . We call a ground instance of a rule a *ground rule*.

Given two sets of ground terms  $E, F$  and a rule  $l \rightarrow r$ , we write  $E \rightarrow_{l \rightarrow r} F$  iff  $F = E \cup \{r\}$  and  $l \subseteq E$ , where  $l$  is a set of terms. We write  $E \rightarrow F$  iff there exists rule  $l \rightarrow r$  such that  $E \rightarrow_{l \rightarrow r} F$ .

**Definition 2.3.** A *derivation*  $D$  of length  $n \geq 0$  is a sequence of finite sets of ground terms  $E_0, E_1, \dots, E_n$  such that  $E_0 \rightarrow E_1 \rightarrow \dots \rightarrow E_n$ , where

$E_i = E_{i-1} \cup \{t_i\}, \forall i = \{1, \dots, n\}$ . A term  $t$  is *derivable* from a set of terms  $E$  iff there exists a derivation  $D = E_0, \dots, E_n$  such that  $E_0 = E$  and  $t \in E_n$ . A set of terms  $T$  is derivable from  $E$ , iff every  $t \in T$  is derivable from  $E$ . We denote  $\text{Der}(E)$  set of terms derivable from  $E$ .

Local intruder  $I$  can send a message  $m$ , if  $m \in \text{Der}(K_I)$ , where  $K_I$  is a current knowledge of intruder  $I$ .

*Protocol session execution.* Now, we can present a course of a protocol execution. We first introduce a notion of *symbolic* execution, where data exchanged among the agents and intruders are not instantiated and represented as (possibly non-ground) terms. This execution is constrained by some conditions. Whenever these conditions are satisfied by an appropriate ground instantiation of variables, we obtain a *concrete* execution, or simply an *execution*. These conditions are defined by *constraint systems*:

**Definition 2.4.** Let  $E$  be a set of terms and  $t$  be a term, we define the couple  $(E, t)$  denoted  $E \triangleright t$  to be a *constraint*. A *constraint system* is a set

$$\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$$

where  $n$  is an integer and  $E_i \triangleright t_i$  is a constraint for  $i \in \{1, \dots, n\}$ .

We extend the definition of  $\text{Vars}(\cdot)$  to constraint system  $\mathcal{S}$  in a natural way. We say that  $\mathcal{S}$  is normalized, if every term in  $\mathcal{S}$  is normalized. By  $\ulcorner \mathcal{S} \urcorner$  we will denote a constraint system  $\{\ulcorner E_i \urcorner \triangleright \ulcorner t_i \urcorner\}_{i=1, \dots, n}$ .

**Definition 2.5.** A ground substitution  $\sigma$  is a *model* of constraint  $E \triangleright t$  (or  $\sigma$  satisfies this constraint), if  $\ulcorner t\sigma \urcorner \in \text{Der}(\ulcorner E\sigma \urcorner)$ . A ground substitution  $\sigma$  is a *model* of a constraint system  $\mathcal{S}$ , if it satisfies all the constraints of  $\mathcal{S}$  and  $\text{dom}(\sigma) = \text{Vars}(\mathcal{S})$ .

**Definition 2.6.** A configuration  $\Pi$  of a protocol session is a quadruple  $\langle PS, \mathcal{K}, \mathcal{Q}, \mathcal{S} \rangle$ , where  $\mathcal{K} = \{\langle I_i, K_i \rangle\}_{i=1, \dots, N}$  represents current knowledges of intruders, and  $\mathcal{Q} = \{\langle c, m_c \rangle\}_{c \in \mathbb{C}}$  is a configuration of channels: for every channel  $c$  queue of messages  $m_c$  is given.

*Transitions* on configurations are defined in Table<sup>3</sup> 1 and will be explained later. Transitions are written in form  $\Pi_1 \xrightarrow{\text{cond}} \Pi_2$  and state that configuration  $\Pi_1$  can evolve to a new configuration  $\Pi_2$  if condition *cond* is satisfied.

---

<sup>3</sup> $\uplus$  represents the union of two disjoint sets:  $A \uplus B = A \cup B$  iff  $A \cap B = \emptyset$ .



1.	$\langle \{ \langle a, (?_f r).l_a \rangle \} \uplus PS, \{ \langle I, K \rangle \} \uplus \mathcal{K}, \mathcal{Q}, \mathcal{S} \rangle \xrightarrow{\iota(f \rightarrow a)=I} \langle \{ \langle a, l_a \rangle \} \cup PS, \{ \langle I, K \rangle \} \cup \mathcal{K}, \mathcal{Q}, \mathcal{S} \cup \{ K \triangleright r \} \rangle$
2.	$\langle \{ \langle a, (!_t s).l_a \rangle \} \uplus PS, \{ \langle I, K \rangle \} \uplus \mathcal{K}, \mathcal{Q}, \mathcal{S} \rangle \xrightarrow{\iota(a \rightarrow t)=I} \langle \{ \langle a, l_a \rangle \} \cup PS, \{ \langle I, K \cup s \rangle \} \cup \mathcal{K}, \mathcal{Q}, \mathcal{S} \rangle$
3.	$\langle \{ \langle a, (!_t s).l_a \rangle \} \uplus PS, \mathcal{K}, \{ \langle a \rightarrow t, m_{a \rightarrow t} \rangle \} \uplus \mathcal{Q}, \mathcal{S} \rangle \xrightarrow{\iota(a \rightarrow t)=\emptyset} \langle \{ \langle a, l_a \rangle \} \cup PS, \mathcal{K}, \{ \langle a \rightarrow t, m_{a \rightarrow t}.s \rangle \} \cup \mathcal{Q}, \mathcal{S} \rangle$
4.	$\langle \{ \langle a, (?_f r).l_a \rangle \} \uplus PS, \mathcal{K}, \{ \langle f \rightarrow a, s.m_{f \rightarrow a} \rangle \} \uplus \mathcal{Q}, \mathcal{S} \rangle \xrightarrow{\iota(f \rightarrow a)=\emptyset} \langle \{ \langle a, l_a \rangle \} \cup PS, \mathcal{K}, \{ \langle f \rightarrow a, m_{f \rightarrow a} \rangle \} \cup \mathcal{Q}, \mathcal{S} \cup \{ \{ \text{enc}(s, k) \} \triangleright \text{enc}(r, k) \} \rangle$

Table 1: Configuration transitions

**Definition 2.7.** A *symbolic execution*  $E_{PS}^S$  of protocol session  $PS$  (with intruders layout  $\iota$ ) is a sequence of configurations obtained by application of transitions to the initial configuration  $\langle PS, \{ \langle I, K_I^0 \rangle \}_{I \in \mathbb{I}}, \{ \langle c, \emptyset \rangle \}_{c \in \mathcal{C}}, \emptyset \rangle$ .

For a substitution  $\sigma$  and a configuration  $\Pi = \langle \{ \langle a_i, l_i \rangle \}_{i=1, \dots, k}, \{ \langle I_i, K_i \rangle \}_{i=1, \dots, N}, \{ \langle c, m_c \rangle \}_{c \in \mathcal{C}}, \{ E_i \triangleright t_i \}_{i=1, \dots, n} \rangle$  we define  $\Pi\sigma$  as  $\langle \{ \langle a_i, l_i \sigma \rangle \}_{i=1, \dots, k}, \{ \langle I_i, K_i \sigma \rangle \}_{i=1, \dots, N}, \{ \langle c, m_c \sigma \rangle \}_{c \in \mathcal{C}}, \{ E_i \sigma \triangleright t_i \sigma \}_{i=1, \dots, n} \rangle$ , where substitutions are applied to lists elementwise.

**Definition 2.8.** An *execution*  $E_{PS} = \{ C_i \sigma \}_{i=1, \dots, m}$  is an instance of a symbolic execution  $\{ C_i \}_{i=1, \dots, m}$  (where  $C_i = \langle PS_i, \mathcal{K}_i, \mathcal{Q}_i, \mathcal{S}_i \rangle$ ) such that all terms of  $C_i \sigma$  are ground and  $\mathcal{S}_m$  is satisfied by  $\sigma$ .

Now we describe the transitions of Table 1. Transition 1 expresses the possibility of intruder  $I$  controlling channel  $f \rightarrow a$  to impersonate  $f$  and send to  $a$  some message compliant with the expected by  $a$  pattern  $r$ , if the current knowledge of  $I$  allows it. An intruder can also intercept messages sent on the channel that he controls (Transition 2). A message sent by an agent on the channel free from intruders is put to the end of the queue of this channel (Transition 3). Transition 4 represents the reading of a message from the queue of the channel.

Let us explain where the constraint  $\{ \text{enc}(s, k) \} \triangleright \text{enc}(r, k)$  comes from in the last transition. Agent  $a$  expects to read a message from the channel compatible with the pattern  $r$ . The first (possibly not yet instantiated) message in the queue is  $s$ . Thus,  $r$  and  $s$  must be unifiable (modulo considered equational theory), and even equivalent when we consider ground instances

of the symbolic executions. Since we will be interested only in concrete executions, but not symbolic, we can use this constraint to express equivalence between  $r$  and  $s$  (Lemma 1).

**Lemma 1.** *For terms  $t_1, t_2$  and substitution  $\sigma$ ,  $\lceil t_1 \sigma \rceil = \lceil t_2 \sigma \rceil$  is true iff  $\sigma$  is a model of  $\{\text{enc}(t_1, k)\} \triangleright \text{enc}(t_2, k)$  for any term  $k$ , i.e.  $\lceil t_1 \sigma \rceil = \lceil t_2 \sigma \rceil$  iff  $\lceil \text{enc}(t_1, k) \sigma \rceil \in \text{Der}(\{\lceil \text{enc}(t_2, k) \sigma \rceil\})$ .*

*Offline communication.* At some point the current knowledge of all local intruders can be shared to derive a secret which probably they cannot deduce separately. In some cases these offline interactions are time-consuming and may be detected. Therefore we consider reasonable that in the intruder strategy modelling they take place after the protocol is over.

*Coordinated attack problem.* Now we can formally state the problem.

*Input:* A finite set of agents  $A$ , a protocol session  $PS = \{\langle a_i, l_i \rangle\}_{i=1, \dots, k}$ , a set of intruders  $\mathbb{I} = \{I_i\}_{i=1, \dots, N}$  each with initial knowledge  $K_{I_i}^0$ , an intruder layout  $\iota$  and some sensitive data given as a finite set of ground terms  $S$ .

*Output:*  $s \in S$  and an execution  $E_{PS}$  of protocol session  $PS$  with its last configuration  $\langle PS, \mathcal{K}, \mathcal{Q}, \mathcal{S} \rangle$  such that  $s \in \text{Der}\left(\bigcup_{\langle K_I, I \rangle \in \mathcal{K}} K_I\right)$ .

### 2.1.2. Solving the problem

We proceed as follows:

1. Guess a sensitive datum  $s$  from  $S$ .
2. Guess a symbolic execution  $E_{PS}^S$  of some length  $\leq \sum_{\langle a, l \rangle \in PS} \text{length}(l) < \infty$ .
3. For the last configuration  $\langle PS, \{\langle K_I, I \rangle\}_{I \in \mathbb{I}}, \mathcal{Q}, \mathcal{S} \rangle$  of  $E_{PS}^S$ , if constraint system  $\mathcal{S} \cup \left\{ \bigcup_{\langle K_I, I \rangle \in \mathcal{K}} K_I \triangleright s \right\}$  is satisfiable with some  $\sigma$ , then the protocol session is insecure and we return  $E_{PS} = E_{PS}^S \sigma$ .

We will show in Sections 3 and 4 that the satisfiability of constraint systems is in  $NP$  in the case of DY+ACI deduction theory.

### 2.2. Attack exploiting XML format of messages

Here we show how to model (using our formalism) attacks based on an XML-representation of messages. A different technique to handle this kind of attacks was presented in [20].

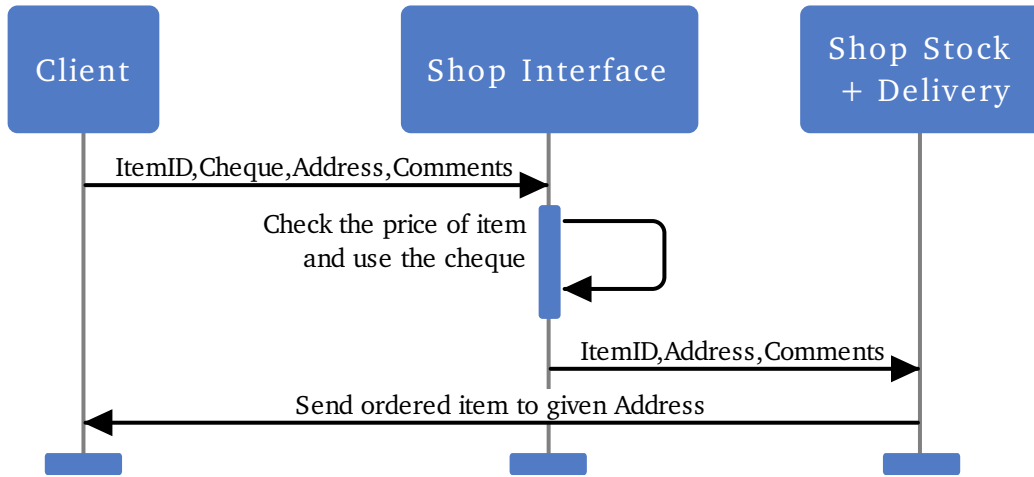


Figure 2: Ordering item scenario

We consider an e-shop that accepts e-cheques, and we suppose that it is presented by a Web Service using SOAP protocol for exchanging messages.

It consists of two services:

- the first exposes the list of goods for sale with their prices and process the orders by accepting payment,
- the second is a delivery service; it receives information from the first one about successfully paid orders, and sends the ordered goods to the buyer.

A simple scenario for ordering item is shown in Figure 2. First, a client sends an order using e-shop interface that consists of an item identifier, e-cheque, delivery address and some comments. Then, the first service of the e-shop checks whether the price of the ordered item corresponds to the received cheque. If it does, the service consumes the cheque and resends the order to the stock/delivery service (without the used e-cheque). Stock and delivery service prepare a parcel with ordered item and send it to given address. The comment is automatically printed on the parcel to give some information to the postman about, for example, delivery time or access instructions.

Suppose, Alice has an e-cheque for 5€. She selected a simple pen (with ItemID simple) to buy, but she liked very much a more expensive gilded one (with ItemID gilded). Can we help Alice to get what she wants for what she has?

Let us formalize the behaviour of scenario players (terms, normalization function and deduction system are defined as in § 3.1.1 except that we will write  $(t_1 \cdot \dots \cdot t_n)$  instead of  $\cdot (\{t_1, \dots, t_n\})$ ). Identifiers starting from a capital letter are considered as variables; numbers and identifier starting from lower-case letter are considered as constants. We model a delivery of item with some *ItemID* to address *Address* with comments *Comments* by the following message:  $\text{sig}((\text{ItemID} \cdot \text{Address} \cdot \text{Comments}), \text{priv}(k_s))$  — a message signed by e-shop, where  $k_s$  its public key, such that no one can produce this message except the shop. We abstract away from the procedure of checking price of the item and will suppose, that Shop Interface expects 5€ e-cheque for Item “*simple*”. For simplicity we assume only two items.

We will use notation for sending and receiving as in § 2.1.

For Shop Interface we have:

$$\begin{aligned} &?_{Client}(\text{simple} \cdot \text{cheque5} \cdot \text{IAddr} \cdot \text{IComm}); \\ &!_{Delivery}(\text{simple} \cdot \text{IAddr} \cdot \text{IComm}). \end{aligned}$$

For Shop Stock/Delivery we have:

$$\begin{aligned} &?_{Interface}(\text{DItemID} \cdot \text{DAddr} \cdot \text{DComm}); \\ &!_{Client} \text{sig}((\text{DItemID} \cdot \text{DAddr} \cdot \text{DComm}), \text{priv}(k_s)). \end{aligned}$$

Alice initially has:

<i>simple, gilded</i> :	identifiers of items;
<i>cheque5</i> :	an e-cheque for 5€;
<i>addr</i> :	her address;
<i>cmnts</i> :	residence digital code;
$k_s$ :	a public key of the shop.

Now we build a mixed constraint system (derivation constraints and equations) to know, whether Alice can do what she wants:

$$\left\{ \begin{array}{l} \{gilded, simple, cheque5, addr, cmnts, k_s\} \triangleright \\ \qquad \qquad \qquad (simple \cdot cheque5 \cdot IAddr \cdot IComm) \end{array} \right\} \quad (1)$$

$$\left\{ \begin{array}{l} (simple \cdot IAddr \cdot IComm) =_{ACI} (DItemID \cdot DAddr \cdot DComm) \\ \{gilded, simple, cheque5, addr, cmnts, k_s, \\ \text{sig}((DItemID \cdot DAddr \cdot DComm), \text{priv}(k_s))\} \triangleright \\ \qquad \qquad \qquad \text{sig}((gilded \cdot addr \cdot DComm), \text{priv}(k_s)) \end{array} \right\} \quad (2)$$

$$\left\{ \begin{array}{l} \{gilded, simple, cheque5, addr, cmnts, k_s\} \triangleright \\ \qquad \qquad \qquad (simple \cdot cheque5 \cdot IAddr \cdot IComm) \end{array} \right\} \quad (3)$$

Constraint (1) shows, that Alice can construct a message expected by the shop from a client. Constraint (2) represents a request from the first to the second service of the shop: left-hand side is a message sent by the interface service, and right-hand side is a message expected by stock/delivery subservice. The last constraint shows, that from the received values Alice can build a message that models a delivery of item with ItemID gilded.

To solve it, we first get rid of syntactic equations by applying most general unifier; and then of equations modulo ACI ( $t_1 =_{ACI} t_2$  is equivalent to  $\lceil t_1 \rceil = \lceil t_2 \rceil$ ) by encoding them into deduction rules (as it was done in § 2.1.2).

Then, one of the solutions is:

$$\begin{array}{llll}
 IAddr & \mapsto addr & IComm & \mapsto (gilded \cdot cmnts) \\
 DItemID & \mapsto gilded & DAddr & \mapsto addr \\
 & & DComm & \mapsto (simple \cdot cmnts)
 \end{array}$$

From this solution we see, that Alice can send a not well-formed comments (that presents two XML-nodes), and Delivery service parser can choose an entry with ID gilded. An attack-request can look like this:

```

<ItemID>simple</ItemID>
<Cheque>cheque5</Cheque>
<Address>addr</Address>
<Comments>cmnts</Comments>
<ItemID>gilded</ItemID>

```

The parser of the first service can return value of the first occurrence of ItemID: `<ItemID>simple</ItemID>`. But the parser of the second one can return `<ItemID>gilded</ItemID>`.

This attack is possible, if Alice constructs a request “by hand”, but a similar attack is probably feasible using XML-injection: Alice when filling a request form enters instead of her comments the following string:

```

cmnts</Comments>
<ItemID>gilded</ItemID><Comments>

```

and in the resulting request we get:

```

<ItemID>simple</ItemID>
<Cheque>cheque5</Cheque>
<Address>addr</Address>
<Comments>cmnts</Comments>

```

```
<ItemID>gilded</ItemID><Comments>
</Comments>
```

This kind of XML-injection attacks was described in [21].

### 3. Satisfiability of general DY+ACI constraint systems

In Section 2 we reduced the problem of protocol insecurity in presence of several intruders to solving a system of deducibility constraints. In this section we present a decision procedure for a constraint system where Dolev-Yao deduction system is extended by an associative-commutative-idempotent symbol (DY+ACI). We consider operators for pairing, symmetric and asymmetric encryptions, decryption, signature and an ACI operator that will be used as a set constructor.

As for the proof structure, after introducing the formal notations, the main steps to show the decidability are as follows:

1. We present an algorithm for solving a ground derivability in DY+ACI model.
2. We prove, that the normalization does not change satisfiability: either we normalize a model or a constraint system.
3. We show existence of a conservative solution of satisfiable constraint system: a substitution  $\sigma$  that sends a variable to an ACI-set of quasi-subterms of the constraint system instantiated with  $\sigma$  together with priv-ed atoms of the constraint system;
4. We give a bound on size of a conservative solution, and, as consequence, we obtain decidability.

#### 3.1. Formal introduction to the problem

##### 3.1.1. Terms and notions

**Definition 3.1.** *Terms* are defined according to the following grammar:

$$\begin{aligned}
term & ::= variable \mid atom \mid \text{pair}(term, term) \mid \\
& \quad \text{enc}(term, term) \mid \cdot(tlist) \mid \text{priv}(Keys) \mid \\
& \quad \text{aenc}(term, Keys) \mid \text{sig}(term, \text{priv}(Keys)) \\
Keys & ::= variable \mid atom \\
tlist & ::= term \mid term, tlist
\end{aligned}$$

where  $atom \in \mathcal{A}$  and  $variable \in \mathcal{X}$ . We denote  $\mathcal{T}(\mathcal{A}, \mathcal{X})$  the set of all terms over a set of atoms  $\mathcal{A}$  and a set of variables  $\mathcal{X}$ . For short, we write  $\mathcal{T}$  instead of  $\mathcal{T}(\mathcal{A}, \mathcal{X})$ .

By  $\text{sig}(p, \text{priv}(a))$  we mean a signature of message  $p$  with private key  $\text{priv}(a)$ . We do not assume that one can retrieve the message itself from the signature.

Note that we do allow complex keys for symmetric encryption only. As a consequence, we have to introduce a condition on substitution applications: substitution  $\sigma$  cannot be applied to the term  $t$ , if after replacing the resulting entity is not a term (for example, we cannot apply  $\sigma = \{x \mapsto \text{pair}(a, b)\}$  to the term  $\text{aenc}(a, x)$ ).

We denote a term on  $i$ -th position of a list  $L$  as  $L[i]$ . Then  $t \in L$  is a shortcut for  $\exists i : t = L[i]$ . We also define two binary relations  $\subseteq$  and  $\approx$  on lists as follows:  $L_1 \subseteq L_2$  if and only if any  $t \in L_1$  implies  $t \in L_2$ ;  $L_1 \approx L_2$  if and only if  $L_1 \subseteq L_2$  and  $L_2 \subseteq L_1$ , and naturally extend them if  $L_1$  or  $L_2$  is a set.

**Definition 3.2.** We consider symbol  $\cdot$  to be associative, commutative, idempotent (shortly, *ACI*).

We will use  $\text{bin}$  throughout the paper as a generalization of all binary operators:  $\text{bin} \in \{\text{enc}, \text{aenc}, \text{pair}, \text{sig}\}$ .

**Definition 3.3.** For every term  $t \in \mathcal{T}$  we define its root symbol by

$$\text{root}(t) = \begin{cases} \text{bin}, & \text{if } t = \text{bin}(p, q) \\ \cdot, & \text{if } t = \cdot(L), \\ \text{priv}, & \text{if } t = \text{priv}(p), \\ t, & \text{if } t \in \mathcal{X} \cup \mathcal{A}, \end{cases}$$

**Definition 3.4.** For any term  $t \in \mathcal{T}$  we define its *set of elements* by:

$$\text{elems}(t) = \begin{cases} \bigcup_{p \in L} \text{elems}(p) & \text{if } t = \cdot(L); \\ \{t\}, & \text{otherwise.} \end{cases}$$

We extend  $\text{elems}()$  to sets of terms or lists of terms  $T$  by  $\text{elems}(T) = \bigcup_{t \in T} \text{elems}(t)$ .

**Example 1.** For term  $t = \cdot(\{a, \cdot(\{b, a, \text{pair}(a, b)\}), \text{pair}(\cdot(\{b, b\}), a)\})$  set of its elements is  $\text{elems}(t) = \{a, b, \text{pair}(\cdot(\{b, b\}), a), \text{pair}(a, b)\}$ .

**Definition 3.5.** Let  $\prec$  be a strict total order on  $\mathcal{T}$ , such that comparing can be done in polynomial time.

**Definition 3.6.** The cardinality of a set  $P$  is denoted by  $|P|$ .

**Definition 3.7.** The *normal form* of a term  $t$  (denoted by  $\lceil t \rceil$ ) is recursively defined by:

- $\lceil t \rceil = t$ , if  $t \in \mathcal{X} \cup \mathcal{A}$
- $\lceil \text{bin}(t_1, t_2) \rceil = \text{bin}(\lceil t_1 \rceil, \lceil t_2 \rceil)$
- $\lceil \text{priv}(t) \rceil = \text{priv}(\lceil t \rceil)$
- $\lceil \cdot(L) \rceil = \begin{cases} \cdot(L'), & \text{if } |\lceil \text{elems}(L) \rceil| > 1 \text{ and } L' \approx \lceil \text{elems}(L) \rceil \\ & \text{and for all } i < j, L'[i] \prec L'[j]; \\ t', & \text{if } \lceil \text{elems}(L) \rceil = \{t'\} \end{cases},$

where for set of terms  $T$ ,  $\lceil T \rceil = \{\lceil t \rceil : t \in T\}$ .

We can show easily that two terms are congruent modulo the ACI properties of  $\cdot$  iff they have the same normal form. Other properties are stated in Lemma 4.

**Example 2.** Referring to Example 1 for the value of term  $t$ , we have  $\lceil t \rceil = \cdot(\{a, b, \text{pair}(a, b), \text{pair}(b, a)\})$ .

**Definition 3.8.** Let  $t$  be a term. We define a set of quasi-subterms  $\text{QSub}(t)$  as follows:

$$\text{QSub}(t) = \begin{cases} \{t\}, & \text{if } t \in \mathcal{X} \cup \mathcal{A}; \\ \{t\} \cup \text{QSub}(t_1), & \text{if } t = \text{priv}(t_1); \\ \{t\} \cup \text{QSub}(t_1) \cup \text{QSub}(t_2), & \text{if } t = \text{bin}(t_1, t_2); \\ \{t\} \cup \bigcup_{p \in \text{elems}(L)} \text{QSub}(p), & \text{if } t = \cdot(L) \end{cases}$$

If  $T$  — set of terms, then  $\text{QSub}(T) = \bigcup_{t \in T} \text{QSub}(t)$ . If  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$  is a constraint system, we define  $\text{QSub}(\mathcal{S}) = \bigcup_{t \in \bigcup_{i=1}^n E_i \cup \{t_i\}} \text{QSub}(t)$ .

**Example 3.** Referring to Example 1, we have

$$\text{QSub}(t) = \{ \cdot(\{a, \cdot(\{b, a, \text{pair}(a, b)\}), \text{pair}(\cdot(\{b, b\}), a)\}), \\ a, b, \text{pair}(a, b), \text{pair}(\cdot(\{b, b\}), a), \cdot(\{b, b\}) \}.$$



**Definition 3.9.** Let  $t$  be a term. We define  $\text{Vars}(t)$  as set of all the variables in  $t$ :

$$\text{Vars}(t) = \mathcal{X} \cap \text{Sub}(t)$$

We define  $\text{Sub}(t)$  as the set of subterms of  $t$  and the DAG-size of a term, as the number of its different subterms. The DAG-size gives the size of a natural representation of a term in the considered ACI theory.

**Definition 3.10.** Let  $t$  be a term. We define  $\text{Sub}(t)$  as follows:

$$\text{Sub}(t) = \begin{cases} \{t\}, & \text{if } t \in \mathcal{X} \cup \mathcal{A}; \\ \{t\} \cup \text{Sub}(t_1), & \text{if } t = \text{priv}(t_1); \\ \{t\} \cup \text{Sub}(t_1) \cup \text{Sub}(t_2), & \text{if } t = \text{bin}(t_1, t_2); \\ \{t\} \cup \bigcup_{p \in L} \text{Sub}(p), & \text{if } t = \cdot(L). \end{cases}$$

If  $T$  is a set of terms, then  $\text{Sub}(T) = \bigcup_{t \in T} \text{Sub}(t)$ . If  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$  is a constraint system, we define  $\text{Sub}(\mathcal{S}) = \bigcup_{t \in \bigcup_{i=1}^n E_i \cup \{t_i\}} \text{Sub}(t)$ .

**Example 4.** Referring to Example 1, we have

$$\begin{aligned} \text{Sub}(t) = \{ & \cdot(\{a, \cdot(\{b, a, \text{pair}(a, b)\}), \text{pair}(\cdot(\{b, b\}), a)\}), \\ & \cdot(\{b, a, \text{pair}(a, b)\}), \text{pair}(\cdot(\{b, b\}), a), \\ & a, b, \text{pair}(a, b), \cdot(\{b, b\}) \}. \end{aligned}$$

**Definition 3.11.** We define a DAG-size  $\text{size}_{\text{DAG}}$  of a term  $t$  as  $\text{size}_{\text{DAG}}(t) = |\text{Sub}(t)|$ , for set of terms  $T$ ,  $\text{size}_{\text{DAG}}(T) = |\text{Sub}(T)|$  and for constraint system  $\mathcal{S}$  as  $\text{size}_{\text{DAG}}(\mathcal{S}) = |\text{Sub}(\mathcal{S})|$ .

Remark, that for a constraint system such a definition does not polynomially approximate a number of bits needed to write it down(cf. Def. 4.1).

We define a Dolev-Yao deduction system modulo ACI equational theory (denoted DY+ACI). It consists of composition rules and decomposition rules, depicted in Table 2 where  $t_1, t_2, \dots, t_m \in \mathcal{T}$ .

We suppose, hereinafter, that for a constraint system  $\mathcal{S}$ ,  $\text{QSub}(\mathcal{S}) \cap \mathcal{A} \neq \emptyset$ . Otherwise, we can add one constraint  $\{a\} \triangleright a$  to  $\mathcal{S}$  which will be satisfied by any substitution. We denote  $\{\text{priv}(t) : t \in T\}$  for set of terms  $T$  as  $\text{priv}(T)$ . We define  $\text{Vars}(\mathcal{S}) = \bigcup_{i=1}^n \text{Vars}(E_i) \cup \text{Vars}(t_i)$ . We say that  $\mathcal{S}$  is normalized, iff for all  $t \in \text{QSub}(\mathcal{S})$ ,  $t$  is normalized.

Composition rules	Decomposition rules
$t_1, t_2 \rightarrow \ulcorner \text{enc}(t_1, t_2) \urcorner$	$\text{enc}(t_1, t_2), \ulcorner t_2 \urcorner \rightarrow \ulcorner t_1 \urcorner$
$t_1, t_2 \rightarrow \ulcorner \text{aenc}(t_1, t_2) \urcorner$	$\text{aenc}(t_1, t_2), \ulcorner \text{priv}(t_2) \urcorner \rightarrow \ulcorner t_1 \urcorner$
$t_1, t_2 \rightarrow \ulcorner \text{pair}(t_1, t_2) \urcorner$	$\text{pair}(t_1, t_2) \rightarrow \ulcorner t_1 \urcorner$
$t_1, \text{priv}(t_2) \rightarrow \ulcorner \text{sig}(t_1, \text{priv}(t_2)) \urcorner$	$\text{pair}(t_1, t_2) \rightarrow \ulcorner t_2 \urcorner$
$t_1, \dots, t_m \rightarrow \ulcorner \cdot(t_1, \dots, t_m) \urcorner$	$\cdot(t_1, \dots, t_m) \rightarrow \ulcorner t_i \urcorner \text{ for all } i$

Table 2: DY+ACI deduction system rules

**Example 5.** We give a sample of general constraint system and its solution within DY+ACI deduction system.

$$\mathcal{S} = \left\{ \begin{array}{ll} \text{enc}(x, a), \text{pair}(c, a) & \triangleright b \\ \cdot(\{x, c\}) & \triangleright a \end{array} \right\},$$

where  $a, b, c \in \mathcal{A}$  and  $x \in \mathcal{X}$ . One of the eventual models within DY+ACI is  $\sigma = \{x \mapsto \text{enc}(\text{pair}(a, b), c)\}$ .

**Definition 3.12.** Let  $T = \{t_1, \dots, t_k\}$  be a non-empty set of terms. Then we define  $\pi(T)$  as follows:

$$\pi(T) = \ulcorner \cdot(t_1, \dots, t_k) \urcorner$$

Remark:  $\pi(\{t\}) = \ulcorner t \urcorner$ .

**Definition 3.13.** We denote  $\text{QSub}(\mathcal{S}) \setminus \mathcal{X}$  as  $\text{QSub}^\circ(S, \mathcal{X})$  or, for shorter notation,  $\text{QSub}(S)$ .

We introduce a transformation  $\pi(H^{\mathcal{S}, \sigma}(\cdot))$  on ground terms that replaces recursively all binary root symbols such that they are different from all the non-variable quasi-subterms of the constraint system instantiated with its model  $\sigma$ , with ACI symbol  $\cdot$ . Later, we will show, that  $\pi(H(\sigma))$  is also a model of  $\mathcal{S}$ .

**Definition 3.14.** Let us have a constraint system  $\mathcal{S}$  which is satisfiable with model  $\sigma$ . Let us fix some  $\alpha \in (\mathcal{A} \cap \text{QSub}(\mathcal{S}))$ . For given  $\mathcal{S}$  and  $\sigma$  we define a function  $H^{\mathcal{S}, \sigma}(\cdot) : \mathcal{T}_g \rightarrow 2^{\mathcal{T}_g}$  as follows:

$$H^{\mathcal{S},\sigma}(t) = \begin{cases} \{\alpha\}, & \text{if } t \in (\mathcal{A} \setminus \text{QSub}(\mathcal{S})); \\ \{a\}, & \text{if } t = a \in (\mathcal{A} \cap \text{QSub}(\mathcal{S})); \\ \{\text{priv}(\pi(H^{\mathcal{S},\sigma}(t_1)))\}, & \text{if } t = \text{priv}(t_1); \\ \{\text{bin}(\pi(H^{\mathcal{S},\sigma}(t_1)), \pi(H^{\mathcal{S},\sigma}(t_2)))\}, & \text{if } t = \text{bin}(t_1, t_2) \\ & \text{if } \ulcorner t \urcorner \in \ulcorner \text{QSub}(S) \urcorner \sigma \urcorner \\ H^{\mathcal{S},\sigma}(t_1) \cup H^{\mathcal{S},\sigma}(t_2), & \text{if } t = \text{bin}(t_1, t_2) \\ & \text{if } \ulcorner t \urcorner \notin \ulcorner \text{QSub}(S) \urcorner \sigma \urcorner \\ \bigcup_{p \in L} H^{\mathcal{S},\sigma}(p), & \text{if } t = \cdot(L). \end{cases}$$

Henceforward, we will omit parameters and write  $H(\cdot)$  instead of  $H^{\mathcal{S},\sigma}(\cdot)$  for shorter notation.

**Definition 3.15.** We define the superposition of  $\pi(\cdot)$  and  $H(\cdot)$  on a set of terms  $T = \{t_1, \dots, t_k\}$  as follows:  $\pi(H(T)) = \{\pi(H(t)) \mid t \in T\}$ .

**Definition 3.16.** Let  $\theta = \{x_1 \mapsto t_1, \dots, x_k \mapsto t_k\}$  be a substitution. We define  $\pi(H(\theta))$  the substitution  $\{x_1 \mapsto \pi(H(t_1)), \dots, x_k \mapsto \pi(H(t_k))\}$ .

Note, that  $\text{dom}(\pi(H(\theta))) = \text{dom}(\theta)$ .

**Example 6.** We refer to Example 5 and show, that  $\pi(H(\sigma))$  is also a model of  $\mathcal{S}$ .  $\pi(H(\text{enc}(\text{pair}(a, b), c))) = \pi(H(\text{pair}(a, b)) \cup \{c\}) = \pi(\{a\} \cup \{b\} \cup \{c\}) = \cdot(\{a, b, c\})$  (we suppose that  $a \prec b \prec c$ ). One can see, that  $\pi(H(\sigma)) = \{x \mapsto \cdot(\{a, b, c\})\}$  is also a model of  $\mathcal{S}$  within DY+ACI.

### 3.1.2. General properties used in proof

The two following lemmas state simple properties of derivability.

**Lemma 2.** Let  $A, B, C \subseteq \mathcal{T}_g$ . Then if  $A \subseteq \text{Der}(B)$  and  $B \subseteq \text{Der}(C)$  then  $A \subseteq \text{Der}(C)$ .

**Lemma 3.** Let  $A, B, C, D \subseteq \mathcal{T}_g$ . Then if  $A \subseteq \text{Der}(B)$  and  $C \subseteq \text{Der}(D)$  then  $A \cup C \subseteq \text{Der}(B \cup D)$ .

In Lemma 4 we list some auxiliary properties that will be used in main proof.

**Lemma 4.** The following statements are true:

1. For terms  $t, t_1, t_2$ , we have  $\ulcorner (t, t) \urcorner = \ulcorner t \urcorner$ ,  $\ulcorner \cdot (t_1, t_2) \urcorner = \ulcorner \cdot (t_2, t_1) \urcorner$ ,  
 $\ulcorner \cdot (t_1, t_2), t_3 \urcorner = \ulcorner \cdot (t_1, \cdot (t_2, t_3)) \urcorner = \ulcorner \cdot (t_1, t_2, t_3) \urcorner$
2. if  $t$  and  $t\sigma$  are terms, then  $\ulcorner t\sigma \urcorner = \ulcorner \ulcorner t\sigma \urcorner \urcorner = \ulcorner \ulcorner t \urcorner \sigma \urcorner = \ulcorner t \urcorner \ulcorner \sigma \urcorner = \ulcorner \ulcorner t \urcorner \ulcorner \sigma \urcorner \urcorner$
3.  $s \in \text{QSub}(\ulcorner t \urcorner) \implies s = \ulcorner s \urcorner$
4.  $\forall s \in \text{Sub}(\ulcorner t \urcorner) \exists s' \in \text{Sub}(t) : s = \ulcorner s' \urcorner$
5.  $\ulcorner \text{elems}(t) \urcorner = \text{elems}(\ulcorner t \urcorner)$
6.  $\ulcorner \cdot (\ulcorner t_1 \urcorner, \dots, \ulcorner t_m \urcorner) \urcorner = \ulcorner \cdot (t_1, \dots, t_m) \urcorner$ ;  $\pi(T) = \pi(\ulcorner T \urcorner)$
7.  $\text{elems}(\ulcorner \cdot (\ulcorner t_1 \urcorner, \dots, \ulcorner t_m \urcorner) \urcorner) = \text{elems}(\cdot (\ulcorner t_1 \urcorner, \dots, \ulcorner t_m \urcorner)) = \bigcup_{i=1, \dots, m} \text{elems}(\ulcorner t_i \urcorner)$
8.  $H(t) = \bigcup_{p \in \text{elems}(t)} H(p)$ ,
9.  $H(t) = H(\ulcorner t \urcorner)$
10.  $\pi(H(t)) = \pi(H(\ulcorner t \urcorner)) = \ulcorner \pi(H(t)) \urcorner = \ulcorner \pi(H(\ulcorner t \urcorner)) \urcorner$
11.  $\pi(T_1 \cup \dots \cup T_m) = \pi(\{\pi(T_1), \dots, \pi(T_m)\})$
12.  $\text{QSub}(\text{QSub}(t)) = \text{QSub}(t)$
13.  $\text{QSub}(\ulcorner t \urcorner) \subseteq \ulcorner \text{QSub}(t) \urcorner$
14.  $\text{QSub}(t\sigma) \subseteq \text{QSub}(t)\sigma \cup \text{QSub}(\text{Vars}(t)\sigma)$
15.  $\text{Sub}(t\sigma) = \text{Sub}(t)\sigma \cup \text{Sub}(\text{Vars}(t)\sigma)$
16.  $|\ulcorner T \urcorner| \leq |T|$ ,  $|T\sigma| \leq |T|$
17.  $\text{elems}(t) \subseteq \text{QSub}(t) \subseteq \text{Sub}(t)$
18. For term  $t$ ,  $\text{size}_{\text{DAG}}(\ulcorner t \urcorner) \leq \text{size}_{\text{DAG}}(t)$ ;  
for set of terms  $T$ ,  $\text{size}_{\text{DAG}}(\ulcorner T \urcorner) \leq \text{size}_{\text{DAG}}(T)$ ;  
for constraint system  $\mathcal{S}$ ,  $\text{size}_{\text{DAG}}(\ulcorner \mathcal{S} \urcorner) \leq \text{size}_{\text{DAG}}(\mathcal{S})$
19.  $\text{QSub}(\cdot(\{t_1, \dots, t_l\})) \subseteq \{\cdot(\{t_1, \dots, t_l\})\} \cup \text{QSub}(t_1) \dots \cup \text{QSub}(t_l)$
20.  $\forall s \in \text{Sub}(t) \text{ size}_{\text{DAG}}(\ulcorner t\sigma \urcorner) \geq \text{size}_{\text{DAG}}(\ulcorner s\sigma \urcorner)$ .

*Proof.* We will give proofs of several statements. Some other technical proofs are given in [Appendix B.1](#)

**Statement 5:** This statement is trivial, if  $t \neq \cdot(L)$ . Otherwise, let  $t = \cdot(t_1, \dots, t_n)$ .

- if  $\ulcorner \text{elems}(t) \urcorner = \{p\}$ , where  $p \neq \cdot(L_p)$ . Then  $\ulcorner t \urcorner = p$  and then  $\text{elems}(\ulcorner t \urcorner) = \text{elems}(p) = \{p\} = \ulcorner \text{elems}(t) \urcorner$ .
- if  $\ulcorner \text{elems}(t) \urcorner = \{p_1, \dots, p_k\}$ ,  $k > 1$ , where  $p_i \neq \cdot(L_i)$  for all  $i$ . Then  $\ulcorner t \urcorner = \cdot(L)$ , where  $L \approx \{p_1, \dots, p_k\}$ . That means, that  $\text{elems}(\ulcorner t \urcorner) = \bigcup_{p \in \{p_1, \dots, p_k\}} \text{elems}(p) = \{p_1, \dots, p_k\}$ .

**Statement 6:** The first part follows from the definition of normal form and Statement 5. The second one directly follows from the first.

**Statement 9:** By induction on  $\text{size}_{\text{DAG}}(t)$ :

- $\text{size}_{\text{DAG}}(t) = 1$  is possible in the only case:  $t = a \in \mathcal{A}$  and as  $a = \ulcorner a \urcorner$ , the equality is trivial.
- Suppose, that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k > 1$ ),  $H(t) = H(\ulcorner t \urcorner)$  holds.
- Given a term  $t : \text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ . We need to prove that  $H(t) = H(\ulcorner t \urcorner)$ .
  - if  $t = \text{priv}(t_1)$ , then  $H(t) = \{\text{priv}(\pi(H(t_1)))\} =$  (by induction supposition)  $= \{\text{priv}(\pi(H(\ulcorner t_1 \urcorner)))\} = H(\text{priv}(\ulcorner t_1 \urcorner)) = H(\ulcorner t \urcorner)$ .
  - if  $t = \text{bin}(p, q)$  and  $\ulcorner t \urcorner \in \ulcorner \text{QSub}(\mathcal{S}) \sigma \urcorner$ . Then  $H(\ulcorner t \urcorner) = H(\text{bin}(\ulcorner p \urcorner, \ulcorner q \urcorner)) = \{\text{bin}(\pi(H(\ulcorner p \urcorner)), \pi(H(\ulcorner q \urcorner)))\} =$  (by induction supposition)  $= \{\text{bin}(\pi(\ulcorner H(p) \urcorner), \pi(\ulcorner H(q) \urcorner))\} =$  (by Statement 6)  $= \{\text{bin}(\pi(H(p)), \pi(H(q)))\} = H(\text{bin}(p, q))$ .
  - if  $t = \text{bin}(p, q)$  and  $\ulcorner t \urcorner \notin \ulcorner \text{QSub}(\mathcal{S}) \sigma \urcorner$ . Then  $H(t) = H(p) \cup H(q) =$  (by induction)  $= H(\ulcorner p \urcorner) \cup H(\ulcorner q \urcorner) =$  (as  $\ulcorner \text{bin}(\ulcorner p \urcorner, \ulcorner q \urcorner) \urcorner = \ulcorner t \urcorner \notin \ulcorner \text{QSub}(\mathcal{S}) \sigma \urcorner$ )  $= H(\text{bin}(\ulcorner p \urcorner, \ulcorner q \urcorner)) = H(\ulcorner t \urcorner)$ .
  - if  $t = \cdot(L)$ , where  $L = \{t_1, \dots, t_m\}$ . Note first, that as  $t = \cdot(L)$ , we have for all  $s \in \text{elems}(t)$ ,  $\text{size}_{\text{DAG}}(s) < \text{size}_{\text{DAG}}(t)$ . Then, by Statement 8,  $H(t) = \bigcup_{p \in \text{elems}(t)} H(p) =$  (by induction supposition)  $= \bigcup_{p \in \text{elems}(t)} H(\ulcorner p \urcorner)$ . On the other part,  $H(\ulcorner t \urcorner) = \bigcup_{p \in \text{elems}(\ulcorner t \urcorner)} H(p) =$  (by Statement 5)  $= \bigcup_{p \in \ulcorner \text{elems}(t) \urcorner} H(p) = \bigcup_{p \in \text{elems}(t)} H(\ulcorner p \urcorner) = H(t)$ .

**Statement 11:** From definition of  $\pi$  and Statement 5, we obtain that  $\text{elems}(\pi(T_i)) = \ulcorner \text{elems}(T_i) \urcorner$ . Next  $\pi(\{\pi(T_1), \dots, \pi(T_m)\}) = \ulcorner \cdot(L) \urcorner$  (here we use  $\ulcorner \cdot(L) \urcorner$  to capture two cases from definition of normalization at once), where  $L \approx \ulcorner \text{elems}(\{\pi(T_1), \dots, \pi(T_m)\}) \urcorner = \ulcorner \bigcup_{i=1, \dots, m} \ulcorner \text{elems}(T_i) \urcorner \urcorner = \ulcorner \bigcup_{i=1, \dots, m} \text{elems}(T_i) \urcorner$ , while  $\pi(T_1 \cup \dots \cup T_m) = \ulcorner \cdot(L') \urcorner$ , where  $L' \approx \ulcorner \bigcup_{i=1, \dots, m} \text{elems}(T_i) \urcorner$ .

**Statement 13:** By induction on  $\text{size}_{\text{DAG}}(t)$ .

- $\text{size}_{\text{DAG}}(t) = 1$ . Then  $t \in \mathcal{A} \cup \mathcal{X}$ . As  $\text{QSub}(t) = \{t\}$  and  $t = \ulcorner t \urcorner$ , the statement holds.
- Suppose, that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k > 1$ ), the statement is true.
- Given a term  $t : \text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ . Let us consider all possible cases:
  - $t = \text{bin}(t_1, t_2)$ . On the one hand,  $\text{QSub}(t) = \{t\} \cup \text{QSub}(t_1) \cup \text{QSub}(t_2)$ . On the other hand,  $\ulcorner t \urcorner = \text{bin}(\ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner)$  and then,  $\text{QSub}(\ulcorner t \urcorner) = \{\ulcorner t \urcorner\} \cup \text{QSub}(\ulcorner t_1 \urcorner) \cup \text{QSub}(\ulcorner t_2 \urcorner)$ . Then, as  $\text{QSub}(\ulcorner t_i \urcorner) \subseteq \ulcorner \text{QSub}(t_i) \urcorner$ , we have that  $\text{QSub}(\ulcorner t \urcorner) \subseteq \ulcorner \text{QSub}(t) \urcorner$ .
  - $t = \text{priv}(t_1)$ . Proof is similar to one for the case above.
  - $t = \cdot(L)$ . We have  $\text{QSub}(t) = \{t\} \cup \bigcup_{p \in \text{elems}(L)} \text{QSub}(p)$ . From Statement 5 we have  $\text{elems}(\ulcorner \cdot(L) \urcorner) = \ulcorner \text{elems}(\cdot(L)) \urcorner$ , and then,  $\text{QSub}(\ulcorner \cdot(L) \urcorner) = \{\ulcorner \cdot(L) \urcorner\} \cup \bigcup_{p \in \text{elems}(\ulcorner \cdot(L) \urcorner)} \text{QSub}(p) = \ulcorner \{\cdot(L)\} \urcorner \cup \bigcup_{p \in \text{elems}(\cdot(L))} \text{QSub}(\ulcorner p \urcorner) \subseteq$  (by supposition)  $\subseteq \ulcorner \{\cdot(L)\} \urcorner \cup \bigcup_{p \in \text{elems}(\cdot(L))} \ulcorner \text{QSub}(p) \urcorner = \ulcorner \{\cdot(L)\} \urcorner \cup \bigcup_{p \in \text{elems}(\cdot(L))} \text{QSub}(p) \urcorner = \ulcorner \text{QSub}(t) \urcorner$ .

**Statement 14:** By induction on  $\text{size}_{\text{DAG}}(t)$

- $\text{size}_{\text{DAG}}(t) = 1$ .
  - $t \in \mathcal{A}$ . As  $t\sigma = t$  and  $\text{Vars}(t) = \emptyset$ , the statement becomes trivial.
  - $t \in \mathcal{X}$ . Then  $\text{QSub}(t)\sigma = t\sigma$ ,  $\text{Vars}(t) = \{t\}$ ; We have  $\text{QSub}(t\sigma) \subseteq \{t\sigma\} \cup \text{QSub}(t\sigma)$ .
- Suppose, that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k \geq 1$ ), the statement is true.
- Given a term  $t : \text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ . Let us consider all possible cases:
  - $t = \text{bin}(t_1, t_2)$ . Then  $t\sigma = \text{bin}(t_1\sigma, t_2\sigma)$  and  $\text{Vars}(t) = \text{Vars}(t_1) \cup \text{Vars}(t_2)$ .  $\text{QSub}(t\sigma) = \{t\sigma\} \cup \text{QSub}(t_1\sigma) \cup \text{QSub}(t_2\sigma) \subseteq$  (as  $\text{size}_{\text{DAG}}(t_i) < k$ )  $\subseteq \{t\sigma\} \cup \text{QSub}(t_1)\sigma \cup \text{QSub}(\text{Vars}(t_1)\sigma) \cup \text{QSub}(t_2)\sigma \cup \text{QSub}(\text{Vars}(t_2)\sigma) = \{t\sigma\} \cup$

$$\begin{aligned}
& \text{QSub}(t_1)\sigma \cup \text{QSub}(t_2)\sigma \cup \text{QSub}((\text{Vars}(t_1) \cup \text{Vars}(t_2))\sigma) = \\
& \text{QSub}(t)\sigma \cup \text{QSub}(\text{Vars}(t)\sigma). \\
& - t = \text{priv}(t_1). \text{ Proof is similar to one for the case above.} \\
& - t = \cdot(\{t_1, \dots, t_m\}). \text{ We have } t\sigma = \cdot(\{t_1\sigma, \dots, t_m\sigma\}) \text{ and} \\
& \text{Vars}(t) = \bigcup_{i=1, \dots, m} \text{Vars}(t_i). \text{ Then we have } \text{QSub}(t\sigma) = \\
& \{t\sigma\} \cup \bigcup_{p \in \text{elems}(\{t_1\sigma, \dots, t_m\sigma\})} \text{QSub}(p) \subseteq (\text{using Statement 17}) \\
& \subseteq \{t\sigma\} \cup \bigcup_{p \in \bigcup_{i=1}^m \text{QSub}(t_i\sigma)} \text{QSub}(p) = (\text{as } \text{QSub}(\text{QSub}(p)) = \\
& \text{QSub}(p)) = \{t\sigma\} \cup \bigcup_{i=1, \dots, m} \text{QSub}(t_i\sigma) \subseteq (\text{as } \text{size}_{\text{DAG}}(t_i) < \\
& k) \\
& \subseteq \{t\sigma\} \cup \bigcup_{i=1, \dots, m} (\text{QSub}(t_i)\sigma \cup \text{QSub}(\text{Vars}(t_i)\sigma)) = \{t\sigma\} \cup \\
& \bigcup_{i=1, \dots, m} \text{QSub}(t_i)\sigma \cup \text{QSub}\left(\left(\bigcup_{i=1, \dots, m} \text{Vars}(t_i)\right)\sigma\right) \\
& = \text{QSub}(t)\sigma \cup \text{Sub}(\text{Vars}(t)\sigma).
\end{aligned}$$

□

**Lemma 5.** *Given a constraint system  $\mathcal{S}$  and its model  $\sigma$ . Then substitution  $\pi(\text{H}(\sigma))$  is normalized*

*Proof.* For any  $x \in \text{dom}(\pi(\text{H}(\sigma)))$ ,  $x\pi(\text{H}(\sigma)) = \pi(H(x\sigma)) = \ulcorner \pi(H(x\sigma)) \urcorner$  (by Lemma 4). □

**Lemma 6.** *For any normalized term  $t$ ,  $\text{QSub}(t) = \text{Sub}(t)$ .*

*Proof.* By induction on  $\text{size}_{\text{DAG}}(t)$ .

- $\text{size}_{\text{DAG}}(t) = 1$ . Then  $t \in \mathcal{X} \cup \mathcal{A}$ , and thus,  $\text{QSub}(t) = \text{Sub}(t) = \{t\}$ .
- Suppose, that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k > 1$ ),  $\text{QSub}(t) = \text{Sub}(t)$ .
- Given a term  $t : \text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ . We need to show that  $\text{QSub}(t) = \text{Sub}(t)$ .
  - $t = \text{bin}(t_1, t_2)$ . Then  $\text{QSub}(\text{bin}(t_1, t_2)) = \{t\} \cup \text{QSub}(t_1) \cup \text{QSub}(t_2) = (\text{as } \text{size}_{\text{DAG}}(t_i) < k) = \{t\} \cup \text{Sub}(t_1) \cup \text{Sub}(t_2) = \text{Sub}(t)$
  - $t = \text{priv}(t_1)$ . Then  $\text{QSub}(\text{priv}(t_1)) = \{t\} \cup \text{QSub}(t_1) = \{t\} \cup \text{Sub}(t_1) = \text{Sub}(t)$

–  $t = \cdot(L)$ . As  $t$  is normalized, we have that for all  $p \in L$ ,  $p \neq \cdot(L_p)$ . Then  $\text{elems}(L) \approx L$ . Thus, we have  $\text{QSub}(t) = \{t\} \cup \bigcup_{p \in \text{elems}(L)} \text{QSub}(p) = \{t\} \cup \bigcup_{p \in L} \text{Sub}(p) = \text{Sub}(t)$ .

□

In Proposition 1 we remark, that ACI-set of normalized terms has the same deductive expressiveness as that set of normalized terms itself.

**Proposition 1.** *Let  $T$  be a set of terms  $T = \{t_1, \dots, t_k\}$ . Then  $\pi(T) \in \text{Der}(\ulcorner T \urcorner)$  and  $\ulcorner T \urcorner \subseteq \text{Der}(\{\pi(T)\})$ .*

In Proposition 2 we state that a constraint system and its normal form have the same models. In Proposition 3 we show the equivalence, for a constraint system, between the existence of a model and the existence of a normalized model. As a consequence we will need only to consider normalized constraints and models in the sequel.

**Proposition 2.** *The substitution  $\sigma$  is a model of constraint system  $\mathcal{S}$  if and only if  $\sigma$  is a model of  $\ulcorner \mathcal{S} \urcorner$ .*

*Proof.* By definition,  $\sigma$  is a model of  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$ , iff  $\forall i \in \{1, \dots, n\}$ ,  $\ulcorner t_i \sigma \urcorner \in \text{Der}(\ulcorner E_i \sigma \urcorner)$ . But by Lemma 4 we have that  $\ulcorner t_i \sigma \urcorner = \ulcorner \ulcorner t_i \urcorner \sigma \urcorner$  and  $\ulcorner E_i \sigma \urcorner = \ulcorner \ulcorner E_i \urcorner \sigma \urcorner$ . Thus,  $\sigma$  is a model of  $\mathcal{S}$  if and only if  $\sigma$  is a model of  $\ulcorner \mathcal{S} \urcorner$ . □

**Proposition 3.** *The substitution  $\sigma$  is a model of constraint system  $\mathcal{S}$  if and only if  $\ulcorner \sigma \urcorner$  is a model of  $\mathcal{S}$ .*

*Proof.* Proof is similar to one of Proposition 2. □

### 3.2. Ground case of DY+ACI

In Algorithm 2 we need to check whether a ground substitution  $\sigma$  satisfies a constraint system  $\mathcal{S}$ . For this, we have to check the derivability of a ground term from a set of ground terms. In this subsection we present such an algorithm.

First, for the ground case we consider an equivalent to DY+ACI deduction system DY+ACI' obtained from the first by replacing a set of rules

$$\forall i \cdot (t_1, \dots, t_m) \rightarrow \ulcorner t_i \urcorner$$



with

$$\forall s \in \text{elems}(t) \ t \rightarrow \lceil s \rceil, \text{ if } t = \cdot(L).$$

Now, we show an equivalence of the two deduction systems.

**Lemma 7.**  $t \in \text{Der}_{DY+ACI}(E) \iff t \in \text{Der}_{DY+ACI'}(E)$

*Proof sketch.* We show that every rule of one deduction system can be simulated by a combination of rules from the other. It is sufficient to show it for non common rules.

The DY+ACI' rules  $\forall s \in \text{elems}(t) \ t \rightarrow \lceil s \rceil$ , if  $t = \cdot(L)$  are modeled by successive application of rules  $\forall i \ \cdot(t_1, \dots, t_m) \rightarrow \lceil t_i \rceil$ . The converse simulation of  $\cdot(t_1, \dots, t_m) \rightarrow \lceil t_i \rceil$  by DY+ACI' is based on getting all the normalized elements of  $t_i$  and, if  $|\text{elems}(t_i)| \geq 2$  then reconstructing  $\lceil t_i \rceil$  by rule  $p_1, \dots, p_l \rightarrow \lceil \cdot(p_1, \dots, p_l) \rceil$ , where  $p_1, \dots, p_l$  are  $\lceil \text{elems}(t_i) \rceil$ .  $\square$

---

**Algorithm 1:** Verifying derivability of term

---

**Input:** A normalized ground constraint  $E \triangleright t$

**Output:**  $t \in \text{Der}_{DY+ACI}(E)$

---

```

1  Let  $S := \text{QSub}(E) \cup \text{QSub}(t) \setminus E$ ;
2  Let  $D := E$ ;
3  while true do
4      if exists DY rule  $l \rightarrow r$ , such that  $l \subseteq D$  and  $r \in S$  then
5           $S := S \setminus \{r\}$ ;
6           $D := D \cup \{r\}$ ;
7      else
8          if exists  $s \in S : \text{elems}(s) \subseteq D$  then
9               $S := S \setminus \{s\}$ ;
10              $D := D \cup \{s\}$ ;
11         else
12             if exists  $s \in D : \text{elems}(s) \not\subseteq D$  then
13                  $S := S \setminus \text{elems}(s)$ ;
14                  $D := D \cup \text{elems}(s)$ ;
15             else
16                 return  $t \in D$ ;
17             end
18         end
19     end
20 end

```

---

**Lemma 8.** *For Algorithm 1 the following statements are true:*

- *for any step<sup>4</sup>,  $D \cup S = \text{QSub}(E \cup \{t\})$  and  $D \cap S = \emptyset$ ;*
- *it terminates;*
- *for any step,  $D \subseteq \text{Der}_{DY+ACI}(E)$ .*

The following lemmas will be used to prove correctness of the algorithm.

**Lemma 9.**

- *For any decomposition rule  $l \rightarrow r$  of  $DY+ACI'$ , if  $l$  is normalized, then  $r$  is a quasi-subterm of  $l$ .*
- *For any composition rule  $l \rightarrow r$  of  $DY+ACI'$  except  $\{t_1, \dots, t_m\} \rightarrow \ulcorner \cdot (t_1, \dots, t_m) \urcorner$ , if  $l$  is normalized, then  $l \subseteq \text{QSub}(r)$ .*

**Lemma 10.** *After the execution of Step 16 of Algorithm 1, if  $l \rightarrow r$  is a  $DY+ACI'$  rule, such that  $l \subseteq D$  and  $r \notin D$ , then  $l \rightarrow r$  is a composition rule and  $r \notin \text{QSub}(E \cup \{t\})$ .*

*Proof.* Suppose,  $l \rightarrow r$  is a decomposition. By Lemma 9 we have that  $r \in \text{QSub}(l)$  and thus,  $r \in \text{QSub}(D) \subseteq D \cup S$ . Then  $r \notin D$  implies  $r \in S$ , and then, Step 16 must be skipped, as branch 4 or 12 should have been visited.

Thus,  $l \rightarrow r$  is a composition. As algorithm reached Step 16, that means  $r \notin S$  (otherwise one of three branches must be visited and this step would be skipped). As  $r \notin S$  and  $r \notin D$ , we have  $r \notin S \cup D = \text{QSub}(E \cup \{t\})$ .  $\square$

**Lemma 11.** *Given a set of normalized terms  $S$  such that for any  $s \in S$ ,  $\text{elems}(s) \subseteq S$ . Then for any  $DY+ACI'$  composition rule  $l \rightarrow r$  such that  $l \subseteq S$  we have  $\text{elems}(r) \subseteq S \cup \{r\}$ .*

*Proof.* All cases of composition rules except  $t_1, \dots, t_m \rightarrow \ulcorner \cdot (t_1, \dots, t_m) \urcorner$  are trivial, as for them  $\text{elems}(r) = \{r\}$ . For this case, as  $\text{elems}(t_i) \subseteq S$  for all  $i$ , then (by Lemma 4, Statement 7)  $\text{elems}(\ulcorner \cdot (t_1, \dots, t_m) \urcorner) = \text{elems}(\cdot (t_1, \dots, t_m)) = \bigcup_{i=1}^m \text{elems}(t_i) \subseteq S$ .  $\square$

**Proposition 4.** *Algorithm 1 is correct.*

---

<sup>4</sup>Consider two sequential assignments as one step

*Proof.* If algorithm returns true, then, by Lemma 8,  $t \in \text{Der}_{DY+ACI'}(E)$ .

Show, that output is correct, if algorithm returns false. Note, that we consider values of  $D$  and  $S$  that they have after finishing the algorithm. Suppose that output is false ( $t \notin D$ ), but  $t \in \text{Der}_{DY+ACI'}(E)$ . Then there exists minimal by length derivation  $\{E_i\}_{i=0,\dots,n}$  where  $n \geq 1$ ,  $D = E_0$  (as  $D \subseteq \text{Der}_{DY+ACI'}(E)$  and  $t \notin D$ ) and  $t \in E_n$  and  $E_{i+1} \setminus E_i \neq \emptyset$  and  $E_i \rightarrow_{l_i \rightarrow r_i} E_{i+1}$  for all  $i = 0, \dots, n-1$ . Then, applying Lemma 10 we have  $l_0 \rightarrow r_0$  is a composition, and  $r_0 \notin \text{QSub}(E \cup \{t\})$ .

Let  $m$  be the smallest index such that there exists  $s \in S = \text{QSub}(E \cup \{t\}) \setminus D$  and  $s \in E_m$ .

Let  $k$  be the minimal integer, such that  $l_k \rightarrow r_k$  is a decomposition.

Show,  $k \leq m$ . Suppose the opposite, then  $s$  is built by a chain of composition rules from  $D$ . If  $l_{m-1} \rightarrow r_{m-1}$  (where  $r_{m-1} = s$ ) is

- a rule in form of  $\{t_1, \dots, t_c\} \rightarrow \ulcorner (t_1, \dots, t_c) \urcorner$ , then  $\text{elems}(s) \neq \{s\}$  (otherwise it contradicts to minimality of the derivation) and from Lemma 11,  $\text{elems}(s) \subseteq E_{m-1}$  ( $m \neq 1$ , otherwise this step would be executed in the algorithm). As  $s \in S$ , then  $\text{elems}(s) \subseteq \text{QSub}(s) \subseteq \text{QSub}(E \cup \{t\})$ . If  $\text{elems}(s) \subseteq D$  then we got contradiction with the fact, that this step would be executed in the algorithm. If there exists  $e \in \text{elems}(s)$  and  $e \notin D$  (that means,  $e \in S$ ), then we get a contradiction with the minimality of  $m$ , as  $e \in S$  was deduced before.
- any other composition rule, then by Lemma 9,  $l_{m-1} \subseteq \text{QSub}(s)$ , and thus,  $l_{m-1} \subseteq D \cup S$ . Similarly to the previous case,  $m \neq 1$  and we get a contradiction with either minimality of  $m$ , or with the fact, that the algorithm would have to add  $s$  into  $D$ .

Note, that this also shows, that decomposition rule is present in derivation.

Show,  $l_k \not\subseteq D$ . Suppose the opposite. Then by Lemma 9, we have  $r_k \subseteq D$  what contradicts to  $E_{k+1} \setminus E_k \neq \emptyset$ . Thus, at least one element from  $l_k$  is not from  $D$ . Let us consider all possible decomposition rules  $l_k \rightarrow r_k$ :

- $\{\text{pair}(t_1, t_2)\} \rightarrow \ulcorner t_1 \urcorner$ . We know, that  $\text{pair}(t_1, t_2)$  is not in  $D$ , thus, it was built by composition. As  $E_i$  are normalized, the only possible way to build by composition  $\text{pair}(t_1, t_2)$  from normalized terms is  $\{t_1, t_2\} \rightarrow \text{pair}(t_1, t_2)$  (other ways, like  $\text{pair}(t_1, t_2), \text{pair}(t_1, t_2) \rightarrow \ulcorner (\{\text{pair}(t_1, t_2), \text{pair}(t_1, t_2)\}) \urcorner$  would contradict the minimality of the derivation). Thus,  $t_1$  was derived before (or was in  $D$ ), i.e.  $t_1 \in E_k$ . That contradicts to  $E_{k+1} \setminus E_k \neq \emptyset$ .

- $\{\text{pair}(t_1, t_2)\} \rightarrow \ulcorner t_2 \urcorner$ . Similar case.
- $\{\text{enc}(t_1, t_2), \ulcorner t_2 \urcorner\} \rightarrow \ulcorner t_1 \urcorner$ . The case where  $\text{enc}(t_1, t_2) \notin D$  has similar explanations as two cases above. Thus,  $\text{enc}(t_1, t_2) \in D$ . That means,  $t_2 \in \text{QSub}(E \cup \{t\})$  and  $t_2 \notin D$ , i.e.  $t_2 \in S$ . This means,  $t_2$  was derived before and  $t_2 \in S$ , what contradicts to  $k \leq m$ .
- $\{\text{aenc}(t_1, t_2), \ulcorner \text{priv}(t_2) \urcorner\} \rightarrow \ulcorner t_1 \urcorner$  is a similar case to previous one. Note, that if  $\text{priv}(t_2)$  is not in  $D$ , that it must be obtained by decomposition.
- $t \rightarrow \ulcorner s \urcorner$ , where  $s \in \text{elems}(t)$  and  $t = \cdot(L)$ . By Lemma 11,  $\text{elems}(t) \subseteq E_k$ , that contradicts minimality of derivation ( $E_{k+1} \setminus E_k \neq \emptyset$ ).

□

### 3.3. Existence of conservative solutions

In this subsection we will show that for any satisfiable constraint system, there exist a model in special form (so called conservative solution). Roughly speaking, a model in this form can be defined per each variable by set of quasi-subterms of the constraint system and set of atoms (also from the constraint system) that must be prived. This will bound a search space for the model (see § 3.4).

First, we show, that on quasi-subterms of constraint system instantiated with its model, the transformation  $\pi(H(\cdot))$  will be a homomorphism modulo normalization.

**Proposition 5.** *Given a normalized constraint system  $\mathcal{S}$  and its normalized model  $\sigma$ . For all  $t \in \text{QSub}(\mathcal{S})$ ,  $\ulcorner t \pi(H(\sigma)) \urcorner = \ulcorner \pi(H(t\sigma)) \urcorner$ .*

*Proof.* We will prove it by induction on  $|\text{Sub}(t)|$ , where  $t$  is normalized.

- Let  $|\text{Sub}(t)| = 1$ . Then:
  - either  $t \in \mathcal{A}$ . In this case  $t \in (\mathcal{A} \cap \text{QSub}(\mathcal{S}))$ , and as  $t\mu = t$  for any substitution  $\mu$ , then  $\pi(H(t\sigma)) = \pi(H(t)) = \pi(\{t\}) = t$  and  $t\pi(H(\sigma)) = t$ . Thus,  $t\pi(H(\sigma)) = \pi(H(t\sigma))$ .
  - or  $t \in \mathcal{X}$ . As  $\sigma$  is a model and  $t \in \text{QSub}(\mathcal{S})$ , we have  $t \in \text{dom}(\sigma)$ , and, by definition,  $t \in \text{dom}(\pi(H(\sigma)))$ . Then, by definition of  $\pi(H(\sigma))$ ,  $t\pi(H(\sigma)) = \pi(H(t\sigma))$ .

- Assume that for some  $k \geq 1$  if  $|\text{Sub}(t)| \leq k$ , then  $\ulcorner t \pi(H(\sigma)) \urcorner = \ulcorner \pi(H(t\sigma)) \urcorner$ .
- Show, that for any  $t$  such that  $|\text{Sub}(t)| \geq k + 1$ , where  $t = \text{bin}(p, q)$  or  $t = \text{priv}(q)$  or  $t = \cdot(t_1, \dots, t_m)$ , but  $|\text{Sub}(p)| \leq k$ ,  $|\text{Sub}(q)| \leq k$  and  $|\text{Sub}(t_i)| \leq k$ , for all  $i \in \{1, \dots, m\}$ , statement  $\ulcorner t \pi(H(\sigma)) \urcorner = \ulcorner \pi(H(t\sigma)) \urcorner$  is still true. We have:
  - either  $t = \text{bin}(p, q)$ . As  $t = \text{bin}(p, q) \in \text{QSub}(\mathcal{S}) \Rightarrow p \in \text{QSub}(\mathcal{S})$  and  $q \in \text{QSub}(\mathcal{S})$ . As  $|\text{Sub}(p)| < |\text{Sub}(t)|$  and from the induction assumption, we have  $\ulcorner p \pi(H(\sigma)) \urcorner = \ulcorner \pi(H(p\sigma)) \urcorner$ . The same holds for  $q$ .  
 Again, since  $\text{bin}(p, q)\sigma \in \text{QSub}(\mathcal{S})\sigma$  (as  $\text{bin}(p, q) \notin \mathcal{X}$  and  $t \in \text{QSub}(\mathcal{S})$ ) we have that  $\ulcorner \pi(H(\text{bin}(p, q)\sigma)) \urcorner = \ulcorner \pi(H(\text{bin}(p\sigma, q\sigma))) \urcorner = \ulcorner \pi(H(\ulcorner \text{bin}(p\sigma, q\sigma) \urcorner)) \urcorner = \ulcorner \pi(H(\text{bin}(\ulcorner p\sigma \urcorner, \ulcorner q\sigma \urcorner))) \urcorner = \ulcorner \pi(\{\text{bin}(\pi(H(\ulcorner p\sigma \urcorner)), \pi(H(\ulcorner q\sigma \urcorner)))\}) \urcorner = \ulcorner \pi(\{\text{bin}(\ulcorner \pi(H(p\sigma)) \urcorner, \ulcorner \pi(H(q\sigma)) \urcorner)\}) \urcorner = \ulcorner \text{bin}(\ulcorner \pi(H(p\sigma)) \urcorner, \ulcorner \pi(H(q\sigma)) \urcorner) \urcorner = \ulcorner \text{bin}(\ulcorner p \pi(H(\sigma)) \urcorner, \ulcorner q \pi(H(\sigma)) \urcorner) \urcorner = \ulcorner \text{bin}(p \pi(H(\sigma)), q \pi(H(\sigma))) \urcorner = \ulcorner \text{bin}(p, q) \pi(H(\sigma)) \urcorner = \ulcorner t \pi(H(\sigma)) \urcorner$ .
  - or  $t = \cdot(t_1, \dots, t_m)$ . As  $t$  is normalized, it implies that for all  $i \in \{1, \dots, m\}$ ,  $t_i$  are not in form of  $\cdot(L_i)$  and then  $t_i \in \text{QSub}(\mathcal{S})$ , and thus, we have  $t_i \in \text{QSub}(\mathcal{S}) \wedge \ulcorner t_i \pi(H(\sigma)) \urcorner = \ulcorner \pi(H(t_i\sigma)) \urcorner$ .  
 $\pi(H(t\sigma)) = \pi(H(\cdot(t_1\sigma, \dots, t_m\sigma))) = \pi(H(t_1\sigma) \cup \dots \cup H(t_m\sigma)) =$  (by Statement 11 of Lemma 4)  
 $= \pi(\{\pi(H(t_1\sigma)), \dots, \pi(H(t_m\sigma))\}) = \pi(\{\ulcorner t_1 \pi(H(\sigma)) \urcorner, \dots, \ulcorner t_m \pi(H(\sigma)) \urcorner\}) = \ulcorner \cdot(\ulcorner t_1 \pi(H(\sigma)) \urcorner, \dots, \ulcorner t_m \pi(H(\sigma)) \urcorner) \urcorner = \ulcorner \cdot(t_1 \pi(H(\sigma)), \dots, t_m \pi(H(\sigma))) \urcorner = \ulcorner (\cdot(t_1, \dots, t_m)) \pi(H(\sigma)) \urcorner = \ulcorner t \pi(H(\sigma)) \urcorner$
  - or  $t = \text{priv}(q)$ . Then  $q \in \text{QSub}(\mathcal{S})$ .  
 $\pi(H(t\sigma)) = \pi(\{\text{priv}(\pi(H(q\sigma)))\}) = \ulcorner \text{priv}(\pi(H(q\sigma))) \urcorner = \ulcorner \text{priv}(q \pi(H(\sigma))) \urcorner = \ulcorner \text{priv}(q) \pi(H(\sigma)) \urcorner = \ulcorner t \pi(H(\sigma)) \urcorner$ .

Thus, the proposition is proven. □

Now we show, that relation of derivability between a term and a set of terms is stable with regard to transformation  $\pi(H(\cdot))$ .

**Lemma 12.** *Given a normalized constraint system  $\mathcal{S}$  and its normalized model  $\sigma$ . For any DY+ACI rule  $l_1, \dots, l_k \rightarrow r$ ,  $\pi(H(r)) \in \text{Der}(\{\pi(H(l_1)), \dots, \pi(H(l_k))\})$ .*

*Proof idea.* We proceed by considering all possible deduction rules. To give an idea, we show a proof for only one rule (see full proof in [Appendix B.3](#)):  $\text{aenc}(t_1, t_2), \ulcorner \text{priv}(t_2) \urcorner \rightarrow \ulcorner t_1 \urcorner$ . Here we have to show that  $\pi(H(\ulcorner t_1 \urcorner))$  is derivable from  $\{\pi(H(\text{aenc}(t_1, t_2))), \pi(H(\ulcorner \text{priv}(t_2) \urcorner))\}$ . Consider two cases:

- $\exists u \in \text{QSub}(\mathcal{S})$  such that  $\ulcorner \text{aenc}(t_1, t_2) \urcorner = \ulcorner u\sigma \urcorner$ . Then  $\pi(H(\text{aenc}(t_1, t_2))) = \text{aenc}(\pi(H(t_1)), \pi(H(t_2)))$ , and then  $\pi(H(\ulcorner t_1 \urcorner)) = \pi(H(t_1)) \in \text{Der}(\{\text{aenc}(\pi(H(t_1)), \pi(H(t_2))), \ulcorner \text{priv}(\pi(H(t_2))) \urcorner\})$ . On the other hand,  $\pi(H(\ulcorner \text{priv}(t_2) \urcorner)) = \pi(H(\text{priv}(t_2))) = \pi(\{\text{priv}(\pi(H(t_2)))\}) = \ulcorner \text{priv}(\pi(H(t_2))) \urcorner$ .
- $\nexists u \in \text{QSub}(\mathcal{S})$  such that  $\ulcorner \text{aenc}(t_1, t_2) \urcorner = \ulcorner u\sigma \urcorner$ . Then  $\pi(H(\text{aenc}(t_1, t_2))) = \pi(H(t_1) \cup H(t_2))$ . Using Proposition 1, we have  $\ulcorner H(t_1) \cup H(t_2) \urcorner \subseteq \text{Der}(\{\pi(H(\text{aenc}(t_1, t_2)))\})$ , thus (by Lemma 4)  $\ulcorner H(t_1) \urcorner \subseteq \text{Der}(\{\pi(H(\text{aenc}(t_1, t_2)))\})$ . And then, by Proposition 1 we have that  $\pi(H(t_1)) \in \text{Der}(\ulcorner H(t_1) \urcorner)$ . Therefore, by Lemma 2,  $\pi(H(\ulcorner t_1 \urcorner)) = \pi(H(t_1)) \in \text{Der}(\pi(H(\text{aenc}(t_1, t_2))))$ .

□

Using Proposition 5 and Lemma 12 we will show, that transformation  $\pi(H(\cdot))$  preserves the property of substitution to be a model.

**Theorem 1.** *Given a normalized constraint system  $\mathcal{S}$  and its normalized model  $\sigma$ . Then substitution  $\pi(H(\sigma))$  also satisfies  $\mathcal{S}$ .*

*Proof.* Suppose, without loss of generality,  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$ . Let us take any constraint  $(E \triangleright t) \in \mathcal{S}$ . As  $\sigma$  is a model of  $\mathcal{S}$ , there exists a derivation  $D = \{A_0, \dots, A_k\}$  such that  $A_0 = \ulcorner E\sigma \urcorner$  and  $\ulcorner t\sigma \urcorner \in A_k$ .

By Lemma 12 and Lemma 3 we can easily prove that if  $k > 0$ ,  $\pi(H(A_j)) \subseteq \text{Der}(\pi(H(A_{j-1})))$ ,  $j = 1, \dots, k$ . Then, applying transitivity of  $\text{Der}(\cdot)$  (Lemma 2)  $k$  times, we have that  $\pi(H(A_k)) \subseteq \text{Der}(\pi(H(A_0)))$ . In the case where  $k = 0$ , the statement  $\pi(H(A_k)) \subseteq \text{Der}(\pi(H(A_0)))$  is also true.

Using Proposition 5 we get  $\pi(H(A_0)) = \pi(H(E\sigma)) = \ulcorner E\pi(H(\sigma)) \urcorner$ , as  $E \subseteq \text{QSub}(\mathcal{S})$ . The same for  $t$ :  $\pi(H(t\sigma)) = \ulcorner t\pi(H(\sigma)) \urcorner$ , and as  $\ulcorner t\sigma \urcorner \in A_k$ , we have  $\ulcorner t\pi(H(\sigma)) \urcorner \in \pi(H(A_k))$ . Thus, we have that  $\ulcorner t\pi(H(\sigma)) \urcorner \in \pi(H(A_k)) \subseteq \text{Der}(\pi(H(A_0))) = \text{Der}(\ulcorner E\pi(H(\sigma)) \urcorner)$ , that means  $\pi(H(\sigma))$  satisfies any constraint of  $\mathcal{S}$ .  $\square$

From now till the end of subsection we will study a very useful property of  $\pi(H(\sigma))$ . Proposition 6 and its corollary show, that if constraint system has a normalized model ( $\sigma$ ) which sends different variables to different values, then there exists another normalized model ( $\pi(H(\sigma))$ ) that sends any variable of its domain to an ACI-set of some non-variable quasi-subterms of constraint system instantiated by itself and some private keys built with atoms of the constraint system.

**Lemma 13.** *If  $\ulcorner u\sigma \urcorner = \text{enc}(p, q)$ ,  $\sigma$  is normalized,  $u = \ulcorner u \urcorner$ ,  $u \notin \mathcal{X}$  and  $x\sigma \neq y\sigma, x \neq y$ , then there exists  $s \in \text{QSub}(u)$  such that  $s = \text{enc}(p', q')$  and  $\ulcorner s\sigma \urcorner = \text{enc}(p, q)$ . The similar is true in the case of  $\ulcorner u\sigma \urcorner = \text{pair}(p, q)$ ,  $\ulcorner u\sigma \urcorner = \text{aenc}(p, q)$ ,  $\ulcorner u\sigma \urcorner = \text{sig}(p, q)$  and for  $\ulcorner u\sigma \urcorner = \text{priv}(p)$ .*

*Proof.* As  $u = \ulcorner u \urcorner$  and  $\ulcorner u\sigma \urcorner = \text{enc}(p, q)$ , we have:

- $u$  not in form of  $\cdot(L)$ . Then, as  $u \notin \mathcal{X}$  and  $\ulcorner u\sigma \urcorner = \text{enc}(p, q)$ , we have  $u = \text{enc}(p', q')$  (where  $\ulcorner p'\sigma \urcorner = p$  and  $\ulcorner q'\sigma \urcorner = q$ ). Then we can choose  $s = \text{enc}(p', q') = u \in \text{QSub}(u)$ .
- $u = \cdot(t_1, \dots, t_m)$ ,  $m > 1$ , as  $u = \ulcorner u \urcorner$ . Then, for all  $i$ ,  $t_i$  is either a variable, or  $\text{enc}(p'_i, q'_i)$ . But, as  $x\sigma \neq y\sigma, x \neq y$  and as  $\sigma$  is normalized, we can claim, that  $\{t_1, \dots, t_m\}$  contains at most one variable. Then, as  $m > 1$ , there exists  $i$  such that  $t_i = \text{enc}(p'_i, q'_i)$ . Then by definition of normalization function, and from  $\ulcorner u\sigma \urcorner = \text{enc}(p, q)$  we have, that  $\ulcorner \text{elems}(u\sigma) \urcorner = \{\text{enc}(p, q)\}$  and as  $t_i\sigma$  is an element of  $u\sigma$ , we have  $\ulcorner \text{enc}(p'_i, q'_i)\sigma \urcorner = \text{enc}(p, q)$ . Thus, we can choose  $s = t_i$ , as  $t_i \in \text{QSub}(u)$  and  $t_i = \text{enc}(p'_i, q'_i)$ .

The other cases (pair, priv, etc...) can be proved similarly.  $\square$

**Proposition 6.** *Given a normalized constraint system  $\mathcal{S}$  and its normalized model  $\sigma$  such that for all  $x, y \in \text{dom}(\sigma)$ ,  $x \neq y \implies x\sigma \neq y\sigma$ . Then*

for all  $x \in \text{dom}(\pi(H(\sigma)))$  there exist  $k \in \mathbb{N}$  and  $s_1, \dots, s_k \in \text{QSub}^\circ(S) \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$  such that  $\text{root}(s_i) \neq \cdot$  and  $x \pi(H(\sigma)) = \pi(\{s_1 \pi(H(\sigma)), \dots, s_k \pi(H(\sigma))\})$ .

*Proof.* By definition,  $x \pi(H(\sigma)) = \pi(H(x\sigma))$ . Let us take any  $s \in H(x\sigma)$  (note, that  $s$  is a ground term). Then, by definition of  $H(\cdot)$  we have:

- either  $s \in \mathcal{A}$ . Then, by definition of  $H(\cdot)$ ,  $s \in (\mathcal{A} \cap \text{QSub}(\mathcal{S}))$ . Thus,  $s \pi(H(\sigma)) = s$ ,  $s \in \text{QSub}^\circ(S)$ ,  $s \neq \cdot(L)$ ;
- or  $s = \text{bin}(\pi(H(t_1)), \pi(H(t_2)))$  and there exists  $u \in \text{QSub}^\circ(S)$  such that  $\ulcorner u \sigma \urcorner = \ulcorner \text{bin}(t_1, t_2) \urcorner = \text{bin}(\ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner)$ . As all conditions of Lemma 13 are satisfied, then there exists  $v \in \text{QSub}(u)$  such that  $\ulcorner v \sigma \urcorner = \text{bin}(\ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner)$  and  $v = \text{bin}(p, q)$  and as  $u \in \text{QSub}^\circ(S)$  then  $v \in \text{QSub}^\circ(S)$ . By Proposition 5,  $\ulcorner v \pi(H(\sigma)) \urcorner = \pi(H(v\sigma)) = \pi(H(\ulcorner v \sigma \urcorner)) = \pi(H(\text{bin}(t_1, t_2))) = \pi(\{\text{bin}(\pi(H(t_1)), \pi(H(t_2)))\}) = \text{bin}(\pi(H(t_1)), \pi(H(t_2))) = s$ . That means that there exists  $v \in \text{QSub}^\circ(S)$  such that  $v \neq \cdot(L)$  and  $s = \ulcorner v \pi(H(\sigma)) \urcorner$ .
- or  $s = \text{priv}(\pi(H(t_1)))$ . In this case, as  $s$  is ground,  $\pi(H(t_1))$  must be an atom, moreover, by definition of  $H(\cdot)$ , this atom is from  $(\mathcal{A} \cap \text{QSub}(\mathcal{S}))$ . Therefore,  $s = \text{priv}(a)$ , where  $a \in \mathcal{A} \cap \text{QSub}(\mathcal{S})$  (and of course,  $s \neq \cdot(L)$ ).

Thus, for all  $s \in H(x\sigma)$ , there exists  $v \in (\text{QSub}(\mathcal{S}) \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A}) \setminus \mathcal{X} \mid s = \ulcorner v \pi(H(\sigma)) \urcorner$ . Therefore, as  $x \pi(H(\sigma)) = \pi(H(x\sigma))$ , we have that  $x \pi(H(\sigma)) = \pi(\{\ulcorner s_1 \pi(H(\sigma)) \urcorner, \dots, \ulcorner s_k \pi(H(\sigma)) \urcorner\}) = \pi(\{s_1 \pi(H(\sigma)), \dots, s_k \pi(H(\sigma))\})$ , where  $s_1, \dots, s_k \in \text{QSub}^\circ(S) \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$  and  $s_i \neq \cdot(L), \forall 1 \leq i \leq k$ . That proves the proposition.  $\square$

**Corollary 1.** *Given normalized constraint system  $\mathcal{S}$  and  $\sigma'$  — its normalized model, such that  $x \neq y \implies x\sigma' \neq y\sigma'$ . Then there exists a normalized model  $\sigma$  of  $\mathcal{S}$  such that for all  $x \in \text{dom}(\sigma)$  there exist  $k \in \mathbb{N}$  and  $s_1, \dots, s_k \in \text{QSub}^\circ(S) \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$  such that  $x\sigma = \pi(\{s_1\sigma, \dots, s_k\sigma\})$  and  $s_i \neq s_j$ , if  $i \neq j$ ;  $s_i \neq \cdot(L), \forall i$ .*

Any normalized model with property shown in Corollary 1 we will call *conservative*.



### 3.4. Bounds on conservative solutions

To get a decidability result, we first show an upper bound on size of conservative model and then, by reducing any satisfiable constraint system to one that have conservative model and showing that reduced one is smaller (by size) than original one, we obtain an existence of a model with bounded size for any satisfiable constraint system.

**Lemma 14.** *Given a normalized constraint system  $\mathcal{S}$  and its conservative model  $\sigma$ . Then for all  $x \in \text{Vars}(\mathcal{S})$  we have  $\text{QSub}(x\sigma) \subseteq \ulcorner \text{QSub}(\mathcal{S}) \sigma \urcorner \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$ .*

*Proof.* Given a ground substitution  $\sigma$ , let us define a strict total order on variables:  $x \sqsubset y \iff (\text{size}_{\text{DAG}}(x\sigma) < \text{size}_{\text{DAG}}(y\sigma)) \vee (\text{size}_{\text{DAG}}(x\sigma) = \text{size}_{\text{DAG}}(y\sigma) \wedge x \prec y)$ .

By Proposition 6 for all  $x$   $x\sigma = \pi(\{s_1^x\sigma, \dots, s_{k^x}^x\sigma\})$ , where  $s_i^x \in (\text{QSub}(\mathcal{S}) \setminus \mathcal{X}) \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$  and  $s_i^x \neq \cdot(L)$ .

Let us show that if  $y \in \text{Vars}(s_i^x)$  for some  $i$ , then  $y \sqsubset x$ . Suppose, that  $y \in \text{Vars}(s_i^x)$  and  $x \sqsubset y$ . Then  $\text{size}_{\text{DAG}}(x\sigma) = \text{size}_{\text{DAG}}(\pi(\{s_1^x\sigma, \dots, s_{k^x}^x\sigma\})) = \text{size}_{\text{DAG}}(\ulcorner \cdot (s_1^x\sigma, \dots, s_{k^x}^x\sigma) \urcorner) \geq$  (by Lemma 4)  $\geq \text{size}_{\text{DAG}}(\ulcorner s_i^x\sigma \urcorner) > \text{size}_{\text{DAG}}(\ulcorner y\sigma \urcorner)$ , because we know that  $s_i^x = \text{bin}(p, q)$  or  $s_i^x = \text{priv}(p)$  and  $y \in \text{Vars}(s_i^x)$  (for example, in first case,  $\text{size}_{\text{DAG}}(\ulcorner s_i^x\sigma \urcorner) = \text{size}_{\text{DAG}}(\text{bin}(\ulcorner p\sigma \urcorner, \ulcorner q\sigma \urcorner)) = 1 + \text{size}_{\text{DAG}}(\{\ulcorner p\sigma \urcorner, \ulcorner q\sigma \urcorner\})$  and since  $y \in \text{Vars}(\{p, q\})$ , using Statement 20 of Lemma 4, we get  $\text{size}_{\text{DAG}}(\ulcorner s_i^x\sigma \urcorner) \geq 1 + \text{size}_{\text{DAG}}(\ulcorner y\sigma \urcorner)$  And as  $\text{size}_{\text{DAG}}(\ulcorner y\sigma \urcorner) = \text{size}_{\text{DAG}}(y\sigma)$  That means,  $y \sqsubset x$ . Contradiction.

Now we show by induction main property of this lemma.

- let  $x = \min_{\sqsubset}(\text{Vars}(\mathcal{S}))$ .  
Then  $x\sigma = \pi(\{s_1^x\sigma, \dots, s_{k^x}^x\sigma\}) = \ulcorner \cdot (s_1^x\sigma, \dots, s_{k^x}^x\sigma) \urcorner$  and all  $s_i^x$  are ground (as there does not exists  $y \sqsubset x$ ). Then  $x\sigma = \ulcorner \cdot (s_1^x, \dots, s_{k^x}^x) \urcorner$ . We have that  $\text{QSub}(x\sigma) = \{\ulcorner \cdot (s_1^x, \dots, s_{k^x}^x) \urcorner\} \cup \text{QSub}(s_1^x) \cup \dots \cup \text{QSub}(s_{k^x}^x) \subseteq \ulcorner \text{QSub}(\mathcal{S}) \sigma \urcorner \cup \text{priv}(\mathcal{A} \cap \text{QSub}(\mathcal{S}))$ , as for any  $s \in \text{QSub}(s_i^x)$ ,  $s \in \mathcal{T}_g$  and  $s \in \text{QSub}(\mathcal{S})$  or  $s = \text{priv}(a)$  or  $s = a$ , where  $a \in \text{QSub}(\mathcal{S}) \cap \mathcal{A}$ , therefore  $s = \ulcorner s \urcorner = s\sigma \in \ulcorner \text{QSub}(\mathcal{S}) \sigma \urcorner \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$  and  $\ulcorner \cdot (s_1^x, \dots, s_{k^x}^x) \urcorner = x\sigma \in \ulcorner \text{QSub}(\mathcal{S}) \sigma \urcorner$ .
- Suppose, that for all  $z \sqsubset y$  we have  $\text{QSub}(z\sigma) \subseteq \ulcorner \text{QSub}(\mathcal{S}) \sigma \urcorner \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$ .

- Show, that  $\text{QSub}(y\sigma) \subseteq \text{QSub}(\mathcal{S}\sigma) \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$ . We know that  $y\sigma = \pi(\{s_1^y\sigma, \dots, s_{k_y}^y\sigma\}) = \ulcorner \cdot (s_1^y\sigma, \dots, s_{k_y}^y\sigma) \urcorner$  and for any  $z \in \text{Vars}(s_i^y)$ ,  $z \sqsubset y$ . Then we have  $\text{QSub}(y\sigma) = \{y\sigma\} \cup \text{QSub}(\ulcorner s_1^y\sigma \urcorner) \cup \dots \cup \text{QSub}(\ulcorner s_{k_y}^y\sigma \urcorner)$ . We know that  $y\sigma \in \ulcorner \text{QSub}(\mathcal{S})\sigma \urcorner$ . Let us show that  $\text{QSub}(\ulcorner s_i^y\sigma \urcorner) \subseteq \ulcorner \text{QSub}(\mathcal{S})\sigma \urcorner \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$ . By Lemma 4 we have  $\text{QSub}(\ulcorner s_i^y\sigma \urcorner) \subseteq \ulcorner \text{QSub}(s_i^y\sigma) \urcorner \subseteq \ulcorner \text{QSub}(s_i^y)\sigma \urcorner \cup \text{QSub}(\text{Vars}(s_i^y)\sigma) \urcorner = \ulcorner \text{QSub}(s_i^y)\sigma \urcorner \cup \text{QSub}(\text{Vars}(s_i^y)\sigma)$ . We can see that  $\ulcorner \text{QSub}(s_i^y)\sigma \urcorner \subseteq \ulcorner \text{QSub}(\mathcal{S})\sigma \urcorner \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$  (as  $s_i^y \in \text{QSub}(\mathcal{S}) \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$ ); and by induction supposition and by statement proved above we have  $\text{QSub}(\text{Vars}(s_i^y)\sigma) \subseteq \ulcorner \text{QSub}(\mathcal{S})\sigma \urcorner \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$ . Thus,  $\text{QSub}(y\sigma) \subseteq \ulcorner \text{QSub}(\mathcal{S})\sigma \urcorner \cup \text{priv}(\text{QSub}(\mathcal{S}) \cap \mathcal{A})$ .

□

**Proposition 7.** *For normalized constraint system  $\mathcal{S}$  that have conservative model  $\sigma$ , for any  $x \in \text{Vars}(\mathcal{S})$  we have  $\text{size}_{\text{DAG}}(x\sigma) \leq 2 \times \text{size}_{\text{DAG}}(\mathcal{S})$ .*

*Proof.* As  $|\ulcorner \text{Sub}(\mathcal{S})\sigma \urcorner| \leq |\text{Sub}(\mathcal{S})\sigma| \leq |\text{Sub}(\mathcal{S})| = \text{size}_{\text{DAG}}(\mathcal{S})$ , we have (using the fact that  $\sigma$  is normalized and Lemma 14) that  $|\text{Sub}(x\sigma)| = |\text{QSub}(x\sigma)| \leq |\ulcorner \text{QSub}(\mathcal{S})\sigma \urcorner \cup \text{priv}(\mathcal{A} \cap \text{QSub}(\mathcal{S}))| \leq |\ulcorner \text{QSub}(\mathcal{S})\sigma \urcorner| + |\text{priv}(\mathcal{A} \cap \text{QSub}(\mathcal{S}))| \leq \text{size}_{\text{DAG}}(\mathcal{S}) + |\mathcal{A} \cap \text{QSub}(\mathcal{S})| \leq 2 \times \text{size}_{\text{DAG}}(\mathcal{S})$ ; thus,  $\text{size}_{\text{DAG}}(x\sigma) \leq 2 \times \text{size}_{\text{DAG}}(\mathcal{S})$ . □

From this proposition and Corollary 1 we obtain an existence of bounded model for a normalized constraint system that have a model sending different variables to different values. We will reduce an arbitrary constraint system to already studied case. The target properties are stated in Proposition 8 and Corollary 2.

**Lemma 15.** *Given any constraint system  $\mathcal{S}$  and any substitution  $\theta$  such that  $\text{dom}(\theta) = \text{Vars}(\mathcal{S})$  and  $\text{dom}(\theta)\theta \subseteq \text{dom}(\theta)$ . Then  $\text{size}_{\text{DAG}}(\mathcal{S}\theta) \leq \text{size}_{\text{DAG}}(\mathcal{S})$ .*

*Proof.* From Lemma 4 we obtain  $\text{size}_{\text{DAG}}(\mathcal{S}\theta) = |\text{Sub}(\mathcal{S}\theta)| = |\text{Sub}(\mathcal{S})\theta \cup \text{Sub}(\text{Vars}(\mathcal{S})\theta)|$ , but  $\text{Vars}(\mathcal{S})\theta \subseteq \text{dom}(\theta) = \text{Vars}(\mathcal{S})$  ( $\text{Vars}(\mathcal{S}\sigma)$  consists only of variables), and then  $\text{Sub}(\text{Vars}(\mathcal{S})\theta) = \text{Vars}(\mathcal{S})\theta$ . As  $\text{Vars}(\mathcal{S}) \subseteq \text{Sub}(\mathcal{S})$ , we have  $\text{Sub}(\mathcal{S})\theta \cup \text{Sub}(\text{Vars}(\mathcal{S})\theta) = \text{Sub}(\mathcal{S})\theta$ . Thus,  $\text{size}_{\text{DAG}}(\mathcal{S}\theta) = |\text{Sub}(\mathcal{S})\theta| \leq |\text{Sub}(\mathcal{S})| = \text{size}_{\text{DAG}}(\mathcal{S})$ . □

**Definition 3.17.** Let  $\sigma$  and  $\delta$  be substitutions. Then  $\sigma[\delta]$  is a substitution such that  $\text{dom}(\sigma[\delta]) = \text{dom}(\delta)$  and for all  $x \in \text{dom}(\sigma[\delta])$ ,  $x\sigma[\delta] = (x\delta)\sigma$ .

**Lemma 16.** Let  $\theta$  and  $\sigma$  be substitutions such that  $\text{dom}(\theta)\theta = \text{dom}(\sigma)$ ,  $\text{dom}(\sigma) \subseteq \text{dom}(\theta)$  and  $\sigma$  is ground. Then, for any term  $t$ ,  $(t\theta)\sigma = t\sigma[\theta]$ .

*Proof.* When apply  $\theta$  to  $t$ , every variable  $x$  of  $t$  such that  $x \in \text{dom}(\theta)$  is replaced by  $x\theta$ ; then we apply  $\sigma$  to  $t\theta$ : every variable  $y$  of  $t\theta$  is replaced by  $y\sigma$ , thus, every variable  $x$  from  $\text{dom}(\theta)$  will be replaced to  $(x\theta)\sigma$  (as  $\text{dom}(\theta)\theta = \text{dom}(\sigma)$ ); and no other variables will be replaced (as  $\text{dom}(\sigma) \subseteq \text{dom}(\theta)$ ). Thus, we can see that it is the same as in definition of  $\sigma[\theta]$ .  $\square$

**Proposition 8.** Given any satisfiable constraint system  $\mathcal{S}$ . Then there exists a model  $\sigma$  of  $\mathcal{S}$  such that for any  $x \in \text{dom}(\sigma)$ ,  $\text{size}_{\text{DAG}}(x\sigma) \leq 2 \times \text{size}_{\text{DAG}}(\ulcorner \mathcal{S} \urcorner)$

*Proof idea.* Given a normalized model  $\sigma'$  of  $\mathcal{S}$  we build a substitution  $\theta$  that maps different variables whose  $\sigma'$ -instances are the same to one. In this way we obtain a new constraint system and its normalized model on which we can apply Corollary 1 and get its conservative model  $\sigma''$ , and by applying Proposition 7 we get a bound on size for this model. On the other part, we use Lemma 16 to show that  $\sigma''[\theta]$  is a model of  $\ulcorner \mathcal{S} \urcorner$ . And then, using obtained bound and Lemma 15 show existence of a model with stated property. The detailed proof is given in [Appendix B.4](#)  $\square$

**Corollary 2.** Constraint system  $\mathcal{S}$  is satisfiable if and only if there exists a normalized model of  $\mathcal{S}$  defined on  $\text{Vars}(\mathcal{S})$  which maps a variable to a ground term in  $\mathcal{T}(\mathcal{A} \cap \text{QSub}(\ulcorner \mathcal{S} \urcorner), \emptyset)$  with size not greater than double  $\text{size}_{\text{DAG}}(\mathcal{S})$ .

Using this result, we propose an algorithm of satisfiability of constraint

system (Algorithm 2).

---

**Algorithm 2:** Solving constraint system

---

**Input:** A constraint system  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1,\dots,n}$

**Output:** Model  $\sigma$ , if exists; otherwise  $\perp$

---

```

1 Guess for every variable of  $\mathcal{S}$  a value of ground normalized
  substitution  $\sigma$  with size not greater than  $2 \times \text{size}_{\text{DAG}}(\mathcal{S})$ ;
2 if  $\sigma$  satisfies  $E_i \triangleright t_i$  for all  $i = 1, \dots, n$  then
3   | return  $\sigma$ 
4 else
5   | return  $\perp$ 
6 end

```

---

**Proposition 9.** *Algorithm 2 is correct.*

*Proof.* Let  $\sigma$  be an output of Algorithm 2. Then  $\sigma$  is a ground substitution and  $\sigma$  satisfies all constraints from  $\mathcal{S}'$  and therefore, satisfies all constraints from  $\mathcal{S}$ . This means,  $\sigma$  is a model of  $\mathcal{S}$ .  $\square$

**Proposition 10.** *Algorithm 2 is complete.*

*Proof.* Suppose,  $\mathcal{S}$  is satisfiable. Then, by Corollary 2, there exists a guess of value of ground substitution on every element of  $\text{Vars}(\mathcal{S})$  with size not greater than  $2 \times \text{size}_{\text{DAG}}(\mathcal{S})$  which represents a model  $\sigma$  of  $\mathcal{S}$ . Thus, algorithm 2 will return this  $\sigma$ .  $\square$

#### 4. Complexity analysis

In this section we present complexity classes of proposed algorithms. First, we expose what we use as a representation of constraint systems to justify the selected measure of algorithms inputs. Then, we notice that normalization algorithm is polynomial in time. After that we will show the polynomial complexity of the ground derivability algorithm. And as a consequence of the results given before, we obtain that the proposed algorithm for solving general constraint system within DY+ACI model is in  $NP$ .

To reason about complexity, we have to define a size of its input. For terms and set of terms, we will use  $\text{size}_{\text{DAG}}(\cdot) + |\mathbb{E}(\cdot)|$ , where  $\mathbb{E}(\cdot)$  is a set of edges of DAG-representation of its argument. For system of constraints

$\mathcal{S} = \{E_i \triangleright t_i\}_{i=1,\dots,n}$  we will use  $n \times \text{size}_{\text{DAG}}(\mathcal{S}) + |\mathbb{E}(\mathcal{S})|$ . The justification is given below.

**Definition 4.1.** *DAG-representation* of a constraint system  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1,\dots,n}$  is a tagged graph with labeled edges  $\mathbb{G} = \langle \mathbb{V}, \mathbb{E}, \text{tag} \rangle$  ( $\mathbb{V}$  is a set of vertices and  $\mathbb{E}$  is a set of edges; tag is a tagging function defined on  $\mathbb{V}$ ) such that:

- there exists a bijection  $f : \mathbb{V} \mapsto \text{Sub}(\mathcal{S})$ ;
- $\forall v \in \mathbb{V} \text{ tag}(v) = \langle s, m \rangle$ , where
  - $s = \text{root}(f(v))$ ;
  - $m$  is  $2n$ -bit integer, where  $m[2i-1] = 1 \iff f(v) \in E_i$  and  $m[2i] = 1 \iff f(v) = t_i$ .
- $v_1 \xrightarrow{1} v_2 \in \mathbb{E} \iff \exists p \in \mathcal{T} : (\exists \text{bin} : f(v_1) = \text{bin}(f(v_2), p)) \vee f(v_1) = \text{priv}(f(v_2))$ ;
- $v_1 \xrightarrow{2} v_2 \in \mathbb{E} \iff \exists p \in \mathcal{T} : \exists \text{bin} : f(v_1) = \text{bin}(p, f(v_2))$ ;
- $v_1 \xrightarrow{i} v_2 \in \mathbb{E} \iff f(v_1) = \cdot(L) \wedge L[i] = f(v_2)$ ;

**Example 7.** A constraint system

$$\mathcal{S} = \left\{ \begin{array}{ll} \{\text{enc}(a, x), \text{pair}(b, \text{enc}(a, a)), c\} & \triangleright a \\ \{\text{priv}(b), c\} & \triangleright y \\ \{\text{enc}(\text{sig}(a, \text{priv}(c)), y), \text{aenc}(x, b)\} & \triangleright \text{pair}(\text{enc}(a, x), c) \end{array} \right.$$

will be represented as shown<sup>5</sup> in Figure 3. Nodes of this graph represent an element from  $\text{Sub}(\mathcal{S})$  by indicating its root symbol (first part of its tag) and pointers to the children.

Remark that this representation can be refined, as we know that RHS of a constraint is exactly one term. That is why we could tag a node not with  $2n$  bits but with  $n + \lceil \log(n+1) \rceil$  bits (concerning the second component of the tagging function).

---

<sup>5</sup>Label “1” (resp. “2”) of an edge is represented by a left (resp. right) side of its source node

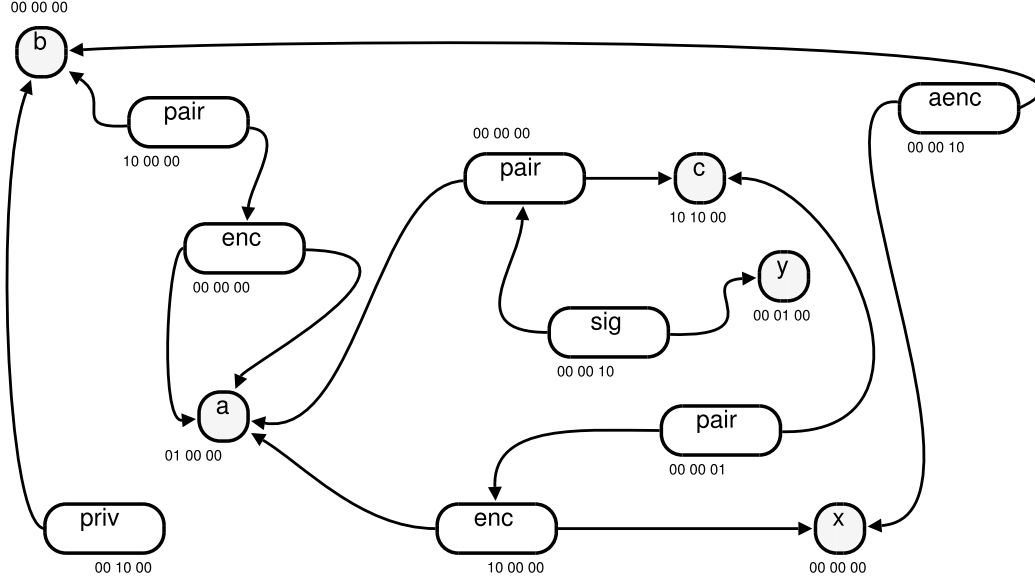


Figure 3: DAG-representation of constraint system  $\mathcal{S}$

The shown representation can be written in not more than  $P(n \times |\mathbb{V}(\mathcal{S})| + |\mathbb{E}(\mathcal{S})|)$  bits of space, where  $\mathbb{V}(\cdot)$  is a set of edges in the DAG-representation, and  $P$  is some polynomial with non-negative coefficients. As we have a bijection between  $\mathbb{V}(\mathcal{S})$  and  $\text{Sub}(\mathcal{S})$ , we obtain  $|\mathbb{V}(\mathcal{S})| = \text{size}_{\text{DAG}}(\mathcal{S})$ . On the other hand, as we are not interested in rigorous estimation of complexity, but work in a polynomial class, we will estimate complexity of algorithms by taking  $n \times \text{size}_{\text{DAG}}(\mathcal{S}) + |\mathbb{E}(\mathcal{S})|$  as the measure of constraint system  $\mathcal{S}$ .

The DAG-representation of a term  $t$  has the similar structure as it was shown for constraint system except that it does not need the second part of a tagging function: we need only  $\text{root}(f(v))$  as a node's tag. The size of this representation will be polynomially bounded by  $\text{size}_{\text{DAG}}(t) + |\mathbb{E}(t)|$ . Thus we give the following definition:

**Definition 4.2.** The *measure* of term  $t$  is defined as:  $\text{measure}(t) = \text{size}_{\text{DAG}}(t) + |\mathbb{E}(t)|$ . For a constraint system  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$ , its measure:  $\text{measure}(\mathcal{S}) = n \times \text{size}_{\text{DAG}}(\mathcal{S}) + |\mathbb{E}(\mathcal{S})|$ .

Note that for the normalized terms and constraint systems, number of edges in their DAG-representation are polynomially limited w.r.t. the number of vertices:

**Lemma 17.** *For any normalized term  $t$ ,  $|\mathbb{E}(t)| < (\text{size}_{\text{DAG}}(t))^2$ . For any normalized constraint system  $\mathcal{S}$ ,  $|\mathbb{E}(\mathcal{S})| < (\text{size}_{\text{DAG}}(\mathcal{S}))^2$ .*

*Proof.* Since the term (resp. constraint system) is normalized, we cannot have more than two edge between two nodes. It evidently holds for binary and unary nodes; for  $\neg$ -nodes it holds because of normalization: if a  $\neg$ -node has two edges to one child, the term is not normalized (one of these edges should have been removed). Therefore, as the graph is directed and acyclic, with as maximum two edges between two nodes, we have not more than  $\text{size}_{\text{DAG}}(x) \times (\text{size}_{\text{DAG}}(x) - 1)$  edges (where  $x$  is a term  $t$  or constraint system  $\mathcal{S}$ ).  $\square$

#### 4.1. Satisfiability of a general DY+ACI constraint systems is in NP

**Lemma 18.** *Given a term  $t$ . Normalization can be done in polynomial time on  $\text{measure}(t)$ . The same holds for a constraint system  $\mathcal{S}$ : normalization can be done in polynomial time on  $\text{measure}(\mathcal{S})$ .*

*Proof idea (for the case of terms).* The algorithm of term normalization works bottom-up by flattening nested ACI-sets, sorting children of ACI-set nodes, merging duplicated nodes while removing unnecessary duplicating edges and removing nodes without incoming edges (except the root-node of  $t$ ).  $\square$

**Proposition 11.** *The general constraint system within DY+ACI satisfiability problem, that Algorithm 2 solves, is in NP.*

*Proof.* Algorithm 2 returns a proof for the decision problem if it exists. We have to show, that the verification of this proof takes a polynomial time with regard to the input problem measure. To do this, we will normalize  $\mathcal{S}\sigma$  and then apply algorithm of checking ground derivability. Using the fact that  $\text{size}_{\text{DAG}}(x\sigma) \leq 2 \times \text{size}_{\text{DAG}}(\mathcal{S})$  and polynomial complexity of the normalization and the ground derivability, we can overapproximate the execution time with polynomial on  $\text{measure}(\mathcal{S})$ . The details of the proof are given in [Appendix B.2](#).  $\square$

On the other hand, we can reuse a technique presented in [14] to show that the satisfiability of a constraint system is an NP-hard problem. The authors encoded 3-SAT problem into an insecurity problem of a single-session sequential protocol. Because the steps of the protocol are linearly ordered, the finding of an attack is reduced to the satisfiability problem of a single constraint system.

**Theorem 2.** *Satisfiability of general DY+ACI constraint systems is NP-complete.*

4.2. *Ground derivability in DY+ACI is in P*

**Proposition 12.** *Algorithm 1 has a polynomial complexity on  $\text{size}_{\text{DAG}}(E \cup \{t\})$ .*

*Proof.* We will give a very coarse estimate.

First remark, that in any step of algorithm,  $|S|$  and  $|D|$  don't exceed  $|\text{QSub}(E \cup \{t\})|$ .

Building  $\text{QSub}(E) \cup \text{QSub}(t)$  takes linear time on  $\text{size}_{\text{DAG}}(E \cup \{t\})$ . Building  $S$  will take not more than  $O(|E| \times |\text{QSub}(E) \cup \text{QSub}(t)|)$ , that is, not more than  $O((\text{size}_{\text{DAG}}(E \cup \{t\}))^2)$ .

The main loop has at most  $|\text{QSub}(E \cup \{t\})| - |E|$  steps. Searching for DY rule with left-hand side in  $D$  and right-hand side in  $S$  is not greater than  $O(|S| \times |D|^2)$  and thus, not greater than  $O((\text{size}_{\text{DAG}}(E \cup \{t\}))^3)$ . The next **if** can be performed in  $O(|S| \times |D| \times (\text{size}_{\text{DAG}}(E \cup \{t\})))$  steps and the last **if** can be also done for cubic time. The check done in **return** statement is linear. And finally, thanks to the Statement 17 of Lemma 4, we can easily justify the claimed complexity.  $\square$

## 5. Satisfiability of general DY constraint system

The previous result on constraint solving for DY+ACI theory can be projected to the classical DY case. We cannot apply it directly, as in the resulting model we will probably have an ACI symbol. Thus, we need to prove the decidability of DY case. The scheme we follow to solve a constraint system within DY deduction system is shown in Figure 4.

First, we can show that if a constraint system is satisfiable within DY, then it is satisfiable within DY+ACI (Proposition 13).

Second, as we know, we can find a model of a given constraint system within DY+ACI.

Third, we will transform the model obtained from previous step (which is in DY+ACI) in such a way, that the resulting substitution will be a model of initial constraint system within DY (Theorem 3). The idea of satisfactory transformation  $\delta$  is simple: we replace any ACI list of terms with nested pairs:  $\cdot(\{t_1, \dots, t_n\})$  we replace with  $\text{pair}(t_1, \text{pair}(\dots, t_n))$ . Note, that this transformation will have a linear complexity and the transformed model will



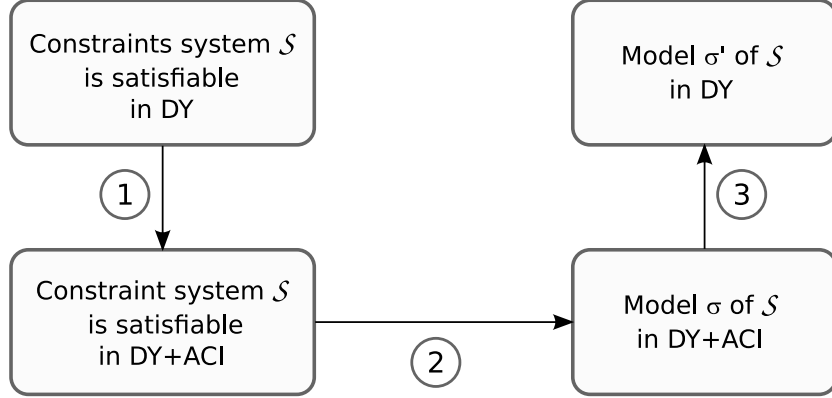


Figure 4: Proof Plan

have the DAG-size not more than twice bigger than initial. This gives us a class of complexity, which is NP, for the problem of satisfiability of general constraint system within DY model.

**Definition 5.1.** We define a replacement  $\delta(t) : \mathcal{T}_g \mapsto \mathcal{T}_g$  in the following way:

$$\delta(t) = \begin{cases} t, & \text{if } t \in \mathcal{X} \cup \mathcal{A}; \\ \text{bin}(\delta(p), \delta(q)), & \text{if } t = \text{bin}(p, q); \\ \text{priv}(\delta(p)), & \text{if } t = \text{priv}(p); \\ \delta(t_1), & \text{if } t = \cdot(t_1); \\ \text{pair}(\delta(t_1), \delta(\cdot(t_2, \dots, t_m))), & \text{if } t = \cdot(t_1, \dots, t_m), m > 1; \end{cases}$$

**Definition 5.2.** Given substitution  $\sigma$ . Then  $\delta(\sigma) = \{x \rightarrow \delta(x\sigma)\}_{x \in \text{dom}(\sigma)}$ . For  $T \subseteq \mathcal{T}_g$ ,  $\delta(T) = \{\delta(t) : t \in T\}$ .

Let us recall classical Dolev-Yao deduction system (DY) in Table 3.

Composition rules	Decomposition rules
$t_1, t_2 \rightarrow \text{enc}(t_1, t_2)$	$\text{enc}(t_1, t_2), t_2 \rightarrow t_1$
$t_1, t_2 \rightarrow \text{aenc}(t_1, t_2)$	$\text{aenc}(t_1, t_2), \text{priv}(t_2) \rightarrow t_1$
$t_1, t_2 \rightarrow \text{pair}(t_1, t_2)$	$\text{pair}(t_1, t_2) \rightarrow t_1$
$t_1, \text{priv}(t_2) \rightarrow \text{sig}(t_1, \text{priv}(t_2))$	$\text{pair}(t_1, t_2) \rightarrow t_2$

Table 3: DY deduction system rules

**Definition 5.3.** A constraint system  $\mathcal{S}$  *standard*, if for all  $s \in \text{Sub}(\mathcal{S})$   $\text{root}(s) \neq \cdot$ . The definition is extended in natural way to terms, sets of terms and substitutions.

We can redefine the notion of derivation for Dolev-Yao deduction system in a natural way, and denote it as  $\text{Der}_{DY}$ .

**Lemma 19.** *Any standard constraint system is normalized.*

**Lemma 20.** *Let  $t$  be a standard term,  $\sigma$  be a normalized substitution. Then  $t\sigma$  is normalized.*

**Proposition 13.** *If a standard constraint system  $\mathcal{S}$  has a model  $\sigma$  within DY deduction system, then  $\mathcal{S}$  has a model within DY+ACI deduction system.*

*Proof.* It is enough to consider the same model  $\sigma$  in DY+ACI. As  $\mathcal{S}\sigma$  is normalized and as DY+ACI includes all the rules from DY, it is easy to show using the same derivation that proves  $\sigma$  to be a model in DY, that  $\sigma$  stays a model of  $\mathcal{S}$  in DY+ACI.  $\square$

The goal of the following reasoning is to show that we can build a model of a constraint system within DY from a model of this constraint system within DY+ACI.

**Lemma 21.** *For any DY+ACI rule  $l_1, \dots, l_k \rightarrow r$ , if  $l_i$  are normalized for all  $i = 1, \dots, k$  then  $\delta(r) \in \text{Der}_{DY}(\{\delta(l_1), \dots, \delta(l_k)\})$ .*

*Proof.* Let us consider all possible rules:

- $t_1, t_2 \rightarrow \ulcorner \text{pair}(t_1, t_2) \urcorner$

As  $t_1$  and  $t_2$  are normalized, then  $\ulcorner \text{pair}(t_1, t_2) \urcorner = \text{pair}(t_1, t_2)$ . We can see, that  $\delta(\text{pair}(t_1, t_2)) = \text{pair}(\delta(t_1), \delta(t_2)) \in \text{Der}_{DY}(\{\delta(t_1), \delta(t_2)\})$ .

- $t_1, t_2 \rightarrow \ulcorner \text{enc}(t_1, t_2) \urcorner$ . Proof of this case can be done by analogy of previous one.
- $t_1, t_2 \rightarrow \ulcorner \text{aenc}(t_1, t_2) \urcorner$ . Proof of this case can be done by analogy of previous one.

- $t_1, \text{priv}(t_2) \rightarrow \ulcorner \text{sig}(t_1, \text{priv}(t_2)) \urcorner$ .

As  $t_1$  and  $\text{priv}(t_2)$  are normalized, then  $\ulcorner \text{sig}(t_1, \text{priv}(t_2)) \urcorner = \text{sig}(t_1, \text{priv}(t_2))$ . We can see, that  $\delta(\text{sig}(t_1, \text{priv}(t_2))) = \text{sig}(\delta(t_1), \delta(\text{priv}(t_2))) = \text{sig}(\delta(t_1), \text{priv}(\delta(t_2))) \in \text{Der}_{DY}(\{\delta(t_1), \text{priv}(\delta(t_2))\})$ , but  $\text{priv}(\delta(t_2)) = \delta(\text{priv}(t_2))$ .

- $t_1, \dots, t_m \rightarrow \ulcorner \cdot(t_1, \dots, t_m) \urcorner$ .

The fact, that  $\text{elems}(\ulcorner \cdot(t_1, \dots, t_m) \urcorner) = \bigcup_{i=1, \dots, m} \text{elems}(t_i)$  follows from  $t_i = \ulcorner t_i \urcorner$  (for all  $i$ ) and Lemma 4.

We can (DY)-derive from  $\{\delta(t_i)\}$  any term in  $\delta(\text{elems}(t_i))$ , trivially, if  $t_i \neq \cdot(L)$  and by applying rules  $\text{pair}(s_1, s_2) \xrightarrow{DY} s_1$  and  $\text{pair}(s_1, s_2) \xrightarrow{DY} s_2$  otherwise (proof by induction on size of  $t_i$ ).

One can observe, that  $\delta(t)$  is a pairing (composition of  $\text{pair}(\cdot, \cdot)$  operator with itself) of  $\delta(\text{elems}(t))$  (by definition of  $\delta(\cdot)$  and normalization function). And then, as  $\delta(t)$  is limited in size, we can (DY)-derive  $\delta(t)$  from  $\delta(\text{elems}(t))$  by iterative use of rule  $s_1, s_2 \xrightarrow{DY} \text{pair}(s_1, s_2)$ , if needed.

Thus, first we can derive  $\delta(\text{elems}(t_i))$  for all  $i$ , and then rebuild (derive with composition rules)  $\delta(\ulcorner \cdot(t_1, \dots, t_m) \urcorner)$ .

- $\text{enc}(t_1, t_2), \ulcorner t_2 \urcorner \rightarrow \ulcorner t_1 \urcorner$ .

As  $\text{enc}(t_1, t_2)$  is normalized, then  $t_1 = \ulcorner t_1 \urcorner$  and  $t_2 = \ulcorner t_2 \urcorner$ . Thus,  $\delta(t_1) \in \text{Der}_{DY}(\{\text{enc}(\delta(t_1), \delta(t_2)), \delta(t_2)\})$  and this is what we need, as  $\delta(\text{enc}(t_1, t_2)) = \text{enc}(\delta(t_1), \delta(t_2))$ .

- $\text{pair}(t_1, t_2) \rightarrow \ulcorner t_1 \urcorner$ . Similar case.
- $\text{pair}(t_1, t_2) \rightarrow \ulcorner t_2 \urcorner$ . Similar case.
- $\text{aenc}(t_1, t_2), \ulcorner \text{priv}(t_2) \urcorner \rightarrow \ulcorner t_1 \urcorner$ . Similar case. Note, that  $\delta(\text{priv}(t_2)) = \text{priv}(\delta(t_2))$ .
- $\cdot(t_1, \dots, t_m) \rightarrow \ulcorner t_i \urcorner$ .

As said above,  $\delta(\text{elems}(\cdot(t_1, \dots, t_m))) \subseteq \text{Der}_{DY}(\{\delta(\cdot(t_1, \dots, t_m))\})$ ; and as  $\delta(\text{elems}(t_i)) \subseteq \delta(\text{elems}(\cdot(t_1, \dots, t_m)))$ , we can (DY)-derive (by composition rules)  $\delta(t_i)$  from  $\delta(\text{elems}(t_i))$ .

□

**Proposition 14.** *Given a standard constraint system  $\mathcal{S}$  and its normalized model  $\sigma$  in DY+ACI. Then, for any subterm of the system  $t \in \text{Sub}(\mathcal{S})$ , we have  $\delta(t\sigma) = t\delta(\sigma)$ .*

*Proof.* The proof is done by induction as in Proposition 5.

- Let  $\text{size}_{\text{DAG}}(t) = 1$ . Then either  $t \in \mathcal{A}$  or  $t \in \mathcal{X}$ . Both are trivial cases.
- Assume that for some  $k \geq 1$  if  $\text{size}_{\text{DAG}}(t) \leq k$ , then  $\delta(t\sigma) = t\delta(\sigma)$ .
- Show, that for  $t$  such that  $\text{size}_{\text{DAG}}(t) \geq k + 1$ , where  $t = \text{bin}(p, q)$  or  $t = \text{priv}(p)$  and  $\text{size}_{\text{DAG}}(p) \leq k$  and  $\text{size}_{\text{DAG}}(q) \leq k$ , statement  $\delta(t\sigma) = t\delta(\sigma)$  is still true. We have:
  - either  $t = \text{bin}(p, q)$ . As  $\delta(\text{bin}(p, q)\sigma) = \delta(\text{bin}(p\sigma, q\sigma)) = \text{bin}(\delta(p\sigma), \delta(q\sigma)) = \text{bin}(p\delta(\sigma), q\delta(\sigma)) = \text{bin}(p, q)\delta(\sigma)$ .
  - or  $t = \text{priv}(p)$ . In this case the proof can be done by analogy with previous one.

Remark: as  $\mathcal{S}$  is standard,  $t \neq \cdot(L)$ .

□

**Theorem 3.** *Given a standard constraint system  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$  and its normalized model  $\sigma$  in DY+ACI. Then  $\delta(\sigma)$  is a model in DY of  $\mathcal{S}$ .*

*Proof.* Let  $E \triangleright t$  be any element of  $\mathcal{S}$ . As  $\sigma$  is a model of  $\mathcal{S}$ , then  $\ulcorner t\sigma \urcorner \in \text{Der}(\ulcorner E\sigma \urcorner)$ . As  $\sigma$  is normalized and  $\mathcal{S}$  is standard, using Lemma 20 we have  $\ulcorner t\sigma \urcorner = t\sigma$  and  $\ulcorner E\sigma \urcorner = E\sigma$ . Then,  $t\sigma \in \text{Der}(E\sigma)$ . That means, there exists a DY+ACI derivation  $D = \{A_0, \dots, A_k\}$  such that  $A_0 = E\sigma$  and  $t\sigma \in A_k$ .

By Lemma 21 and Lemma 3 (which also works for DY case) we can easily prove that if  $k > 0$ ,  $\delta(A_j) \subseteq \text{Der}_{\text{DY}}(\delta(A_{j-1}))$ ,  $j = 1, \dots, k$ . Note, that  $\delta(A)$  is a set of standard terms (and thus, normalized) for any set of terms  $A$ . Then, applying transitivity of  $\text{Der}_{\text{DY}}(\cdot)$  (Lemma 2 for DY)  $k$  times, we have that  $\delta(A_k) \subseteq \text{Der}_{\text{DY}}(\delta(A_0))$ . In the case where  $k = 0$ , the statement  $\delta(A_k) \subseteq \text{Der}_{\text{DY}}(\delta(A_0))$  is also true.

Using Proposition 14 we have that  $\delta(A_0) = \delta(E\sigma) = E\delta(\sigma)$ , as  $E \subseteq \text{QSub}(\mathcal{S})$ . The same for  $t$ :  $\delta(t\sigma) = t\delta(\sigma)$ , and as  $t\sigma \in A_k$ , we have  $t\delta(\sigma) \in \delta(A_k)$ .

Thus, we have that  $t\delta(\sigma) \in \delta(A_k) \subseteq \text{Der}_{DY}(\delta(A_0)) = \text{Der}_{DY}(E\delta(\sigma))$ , that means  $\delta(\sigma)$  DY-satisfies any constraint of  $\mathcal{S}$ .  $\square$

We present an example illustrating the theorem.

**Example 8.** Let us consider a standard constraint system similar to one in Example 5.

$$\mathcal{S} = \left\{ \begin{array}{ll} \text{enc}(x, a), \text{pair}(c, a) & \triangleright b \\ \text{pair}(x, c) & \triangleright a \end{array} \right\},$$

Using Algorithm 2, we can get a model of  $\mathcal{S}$  within DY+ACI, let's say, as in Example 6,  $\sigma = \{x \mapsto \cdot(\{a, b, c\})\}$ .

Then, by applying transformation  $\delta(\cdot)$ , we will get  $\sigma' = \delta(\sigma) = \{x \mapsto \text{pair}(a, \text{pair}(b, c))\}$ . We can see, that  $\sigma'$  is also a model of  $\mathcal{S}$  within DY(as it was proven in Theorem 3).

**Corollary 3** (of Theorem 3 and Proposition 13). *A standard constraint system  $\mathcal{S}$  is satisfiable within DY iff it is satisfiable within DY+ACI.*

**Corollary 4.** *Satisfiability of constraint system within DY is in NP.*

## 6. Conclusions

In this work we presented a decision algorithm of satisfiability of general constraint system within Dolev-Yao deduction system as well as one extended with ACI symbol that can be used to represent sets of terms. The complexity class of the algorithm was proved to be in NP-complete.

We have given also two applications of the presented result: protocol insecurity with non-communicating intruders and discovering XML-based attacks.

## APPENDIX

### Appendix A. General constraints for subterm theories

- composition rules: for all public functional symbols  $f$ ,  $x_1, \dots, x_k \rightarrow f(x_1, \dots, x_k)$
- decomposition rules:  $t_1, \dots, t_m \rightarrow s$ , where  $s$  is a subterm of  $t_i$  for some  $i$ .

We show that the satisfiability of constraint system within subterm deduction system is undecidable in general. More precisely:

**Instance:** a subterm deduction system  $D$ , a constraint system  $C$ .

**Question:** is  $C$  satisfiable ?

To show this, we reduce the halting problem of a Deterministic Turing Machine (TM)  $M$  that works on a single tape. We consider the tape alphabet  $\Gamma = \{0, 1, \flat\}$ , and  $\flat$  is the blank symbol. The states of the TM  $M$  are in a finite set  $Q = \{q_1, q_2, \dots, q_n\}$ . W.l.o.g. we can assume that  $q_1$  (resp.  $q_n$ ) is the unique initial (resp. accepting) state.

In order to represent Turing machine configuration as terms we shall introduce a set of variables  $\mathcal{X}$  and an alphabet  $\mathcal{F}$

$$\mathcal{F} := \{0, 1, \flat, \perp\} \cup Q,$$

where  $\mathcal{F} \setminus \{\perp\}$  are public functional symbols.

The TM configuration with tape  $\perp abcde \perp$ , (where  $\perp$  is an endmarker), with symbol  $d$  under the head, and state  $q$  will be represented by the following term of  $q(c(b(a(\perp), d(e(\perp), x)), x)$  where  $x \in \mathcal{X}$  and  $a, b, c, d, e \in \{0, 1, \flat\}$ .

The composition rules we consider for the TM are  $u \rightarrow f(u)$  for each  $f \in \{0, 1, \flat\}$  and  $u, v, w \rightarrow q(u, v, w)$  for each  $q \in Q$ . For each TM transition of  $M$  we will introduce some decomposition deduction rule that can be applied on a term representation  $q(u, v, q'(u', v', x'))$  iff the transition can be applied to a configuration represented by  $q(u, v, -)$  and generate a configuration represented by  $q'(u', v', -)$ . For each TM instruction of type: “In state  $q$  reading  $a$  go to state  $q'$  and write  $b$ ”, we define the following rule for  $a, b \in \{0, 1, \flat\}$ :

$$q(u, a(v), q'(u, b(v), x)) \rightarrow q'(u, b(v), x)$$

For each instruction of type: “In state  $q$  reading  $a$  go to state  $q'$  and move right”, we define the following rules for  $a \in \{0, 1, \flat\}$ :

$$q(u, a(v), q'(a(u), v, x)) \rightarrow q'(a(u), v, x)$$

A rule is for extending the tape on the right when needed:

$$q(u, \perp, q'(\flat(u), \perp, x)) \rightarrow q'(\flat(u), \perp, x)$$

For each instruction of type: “In state  $q$  reading  $a$  go to state  $q'$  and move left”, we define the following rules for  $a \in \{0, 1, \flat\}$  :

$$q(a(u), v, q'(u, a(v), x)) \rightarrow q'(u, a(v), x)$$

A rule is for extending the tape on the left when needed:

$$q(\perp, a(v), q'(\perp, \flat(a(v)), x)) \rightarrow q'(\perp, \flat(a(v)), x)$$

The resulting deduction system  $D_M$  is obviously a subterm deduction system.

Let us consider a constraint  $\mathcal{S}$  to be solved modulo  $D_M$ :

$$\{q_1(\perp, \perp, x)\} \triangleright q_n(y, z, w)$$

This constraint is satisfiable iff there is a sequence of transitions of  $M$  from a configuration with initial state  $q_1$  and empty tape to a configuration with an accepting state. Hence the constraint solving problem is undecidable.

Let us recall the definition of some properties of constraint systems. These two properties are natural for modeling standard security protocols:

**variable origination:**  $\forall i, \forall x \in \text{Vars}(E_i) \exists j < i \ x \in \text{Vars}(t_j)$ ,

**monotonicity:**  $j < i \implies E_j \subseteq E_i$ .

Note that  $\{\{q_1(\perp, \perp, x)\} \triangleright q_n(y, z, w)\}$  is obviously monotonic.

As a consequence, satisfiability of monotonic constraint systems (but without variable origination) is undecidable. Here is another constraint system, where variable origination is satisfied, but monotony is not. It can be used for reducing the halting problem again:

$$\{\{\perp\} \triangleright x, \{q_1(\perp, \perp, x)\} \triangleright q_n(y, z, w)\}$$

As a consequence, satisfiability of constraint systems with variable origination (but without monotonicity) is undecidable.

We should note by contrast (see [13]), that constraint solving in subterm convergent theories is decidable if the constraint system  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$  satisfies both variable origination and monotonicity.

## Appendix B. Proofs

### Appendix B.1. Proofs of several statements of Lemma 4

**Statement 1:** Follows from the definition of the normalization function and Definition 3.4.

**Statement 3:** By induction on  $\text{size}_{\text{DAG}}(s)$ . Let us fix  $t$ .

- $\text{size}_{\text{DAG}}(s) = \text{size}_{\text{DAG}}(\ulcorner t \urcorner)$ . Then  $s = \ulcorner t \urcorner$  and  $\ulcorner s \urcorner = \ulcorner \ulcorner t \urcorner \urcorner$ , and from Statement 2 (by taking empty  $\sigma$ ) we have  $\ulcorner t \urcorner = \ulcorner \ulcorner t \urcorner \urcorner$ , and thus  $\ulcorner s \urcorner = s$ .
- Suppose, that for some  $k$ , for any  $s \in \text{QSub}(\ulcorner t \urcorner)$ , such that  $\text{size}_{\text{DAG}}(s) > k$ ,  $s = \ulcorner s \urcorner$ .
- Consider case, where  $s \in \text{QSub}(\ulcorner t \urcorner)$  and  $\text{size}_{\text{DAG}}(s) = k$ . Then, by definition of  $\text{QSub}(\cdot)$ ,  $s$  is in
  - $\text{priv}(s) \in \text{QSub}(\ulcorner t \urcorner)$ . By induction supposition we have  $\ulcorner \text{priv}(s) \urcorner = \text{priv}(s)$ , and as  $\ulcorner \text{priv}(s) \urcorner = \text{priv}(\ulcorner s \urcorner)$ , we have  $s = \ulcorner s \urcorner$ .
  - $\text{bin}(s, p) \in \text{QSub}(\ulcorner t \urcorner)$ . By induction we have  $\ulcorner \text{bin}(s, p) \urcorner = \text{bin}(s, p)$ , and as  $\ulcorner \text{bin}(s, p) \urcorner = \text{bin}(\ulcorner s \urcorner, \ulcorner p \urcorner)$ , we have  $s = \ulcorner s \urcorner$ .
  - $\text{bin}(p, s) \in \text{QSub}(\ulcorner t \urcorner)$ . The similar case.
  - $s \in \text{elems}(\cdot(L)), \cdot(L) \in \text{QSub}(\ulcorner t \urcorner)$ . As  $\text{size}_{\text{DAG}}(\cdot(L)) > k$ , we have  $\cdot(L) = \ulcorner \cdot(L) \urcorner$ , that means (from Definition 3.7), that  $L$  is a list of normalized non-ACI-set terms, and as  $\text{elems}(L) \approx L$ , we have that  $s$  is normalized.

**Statement 4:** Suppose the opposite and let us take  $s \in \text{Sub}(\ulcorner t \urcorner)$  with maximal  $\text{size}_{\text{DAG}}(s)$  that does not satisfy the desired property. Note that the “biggest” term in  $\text{Sub}(\ulcorner t \urcorner)$ , i.e.  $\ulcorner t \urcorner$ , does satisfy the property, as we can choose  $s' = t \in \text{Sub}(t)$ . By definition of  $\text{Sub}(\cdot)$  if  $s \in \text{Sub}(\ulcorner t \urcorner)$  and  $s \neq \ulcorner t \urcorner$  then there exists  $r \in \text{Sub}(\ulcorner t \urcorner)$  such that

- $r = \text{bin}(p, s)$  or  $r = \text{bin}(s, p)$  or  $r = \text{priv}(s)$ . Without loss of generality we consider only the first case ( $r = \text{bin}(p, s)$ ) as other ones are similar. As  $\text{size}_{\text{DAG}}(r) > \text{size}_{\text{DAG}}(s)$ , there exists  $r' \in \text{Sub}(t)$  such that  $r = \ulcorner r' \urcorner$ . By definition of  $\ulcorner \cdot \urcorner$ :



- either  $r' = \text{bin}(p', s')$  and  $\ulcorner p' \urcorner = p$  and  $\ulcorner s' \urcorner = s$ . As  $s' \in \text{Sub}(r') \subseteq \text{Sub}(t)$  the property is proved.
- or  $r' = \cdot(L)$  and  $\ulcorner \text{elems}(L) \urcorner = \{r\}$ . Since for all  $q \in \text{elems}(L)$ ,  $\text{root}(q) \neq \cdot$ , then there exists  $q \in \text{elems}(L)$  such that  $q = \text{bin}(p', s')$  and  $\ulcorner p' \urcorner = p$  and  $\ulcorner s' \urcorner = s$ . Using Statement 17 we have  $s' \in \text{Sub}(t)$ .
- $r = \cdot(L)$  and  $s \in L$ . Then, (since  $\text{size}_{\text{DAG}}(r) > \text{size}_{\text{DAG}}(s)$ ) there exists  $r' \in \text{Sub}(t)$  such that  $\ulcorner r' \urcorner = r$ . Using Lemma `reflemma:DAGvsQuasi` and Statement 3 we obtain  $r$  — normalized, and thus,  $\text{root}(s) \neq \cdot$ . Then by definition of  $\ulcorner \cdot \urcorner$  we have  $r' = \cdot(L')$  and  $L \approx \ulcorner \text{elems}(L') \urcorner$ , and thus,  $s \in \ulcorner \text{elems}(L') \urcorner$ , that is there exists  $s' \in \text{elems}(L')$  such that  $s = \ulcorner s' \urcorner$ . Using again Statement 17 we have  $s' \in \text{Sub}(t)$ .

**Statement 7:** To get the first part of equality we first use Statement 5:  $\text{elems}(\ulcorner \cdot(\ulcorner t_1 \urcorner, \dots, \ulcorner t_m \urcorner) \urcorner) = \ulcorner \text{elems}(\cdot(\ulcorner t_1 \urcorner, \dots, \ulcorner t_m \urcorner)) \urcorner$ ; then from Definition 3.4 and Statement 5 we have that  $\text{elems}(\ulcorner \cdot(\ulcorner t_1 \urcorner, \dots, \ulcorner t_m \urcorner) \urcorner)$  is a set of normalized terms. The second part directly follows from Definition 3.4.

**Statement 8:** By induction on  $\text{size}_{\text{DAG}}(t)$ .

- $\text{size}_{\text{DAG}}(t) = 1$ , implies  $t = a \in \mathcal{A}$  and then  $\text{elems}(a) = \{a\}$ , i.e. the equality becomes trivial.
- Suppose, that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k > 1$ ),  $H(t) = \bigcup_{p \in \text{elems}(t)} H(p)$  holds.
- Given a term  $t : \text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ . We should prove  $H(t) = \bigcup_{p \in \text{elems}(t)} H(p)$ .
  - $t = \text{priv}(t_1)$  or  $t = \text{bin}(p, q)$ . In both cases,  $\text{elems}(t) = \{t\}$ , and thus, the equality is trivial.
  - $t = \cdot(L)$ . Note, that for all  $s \in L$ ,  $\text{size}_{\text{DAG}}(s) < k$ . Then, on the one hand,  $H(\cdot(L)) = \bigcup_{p \in L} H(p) =$  (by induction supposition)  $= \bigcup_{p \in L} \bigcup_{p' \in \text{elems}(p)} H(p')$ . On the other hand,  $\bigcup_{p \in \text{elems}(\cdot(L))} H(p) = \bigcup_{p \in \bigcup_{p' \in L} \text{elems}(p')} H(p) = \bigcup_{p' \in L} \bigcup_{p \in \text{elems}(p')} H(p)$ . Thus,  $H(t) = \bigcup_{p \in \text{elems}(t)} H(p)$ .

**Statement 10:** This follows from Statements 9, 6, Definition 3.12 and from equality  $\ulcorner \ulcorner t \urcorner \urcorner = \ulcorner t \urcorner$  (Statement 2).

**Statement 12:**  $\text{QSub}(t) \subseteq \text{QSub}(\text{QSub}(t))$  is trivial as  $t \in \text{QSub}(t)$ . Now we prove by induction on  $\text{size}_{\text{DAG}}(t)$  that  $\text{QSub}(\text{QSub}(t)) \subseteq \text{QSub}(t)$

- $\text{size}_{\text{DAG}}(t) = 1$ . Then  $t \in \mathcal{A} \cup \mathcal{X}$ . As  $\text{QSub}(t) = \{t\}$  the statement is trivial.
- Suppose, that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k \geq 1$ ), the statement is true.
- Given a term  $t : \text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ . Let us consider all possible cases:
  - $t = \text{bin}(t_1, t_2)$ . By definition  $\text{QSub}(t) = \{t\} \cup \text{QSub}(t_1) \cup \text{QSub}(t_2)$ . Then,  $\text{QSub}(\text{QSub}(t)) = \text{QSub}(t) \cup \text{QSub}(\text{QSub}(t_1)) \cup \text{QSub}(\text{QSub}(t_2))$  and, as  $\text{size}_{\text{DAG}}(t_i) < k$  for  $i = 1, 2$ , by using induction supposition we obtain the wanted property.
  - $t = \text{priv}(t_1)$ . Proof is similar to one for the case above.
  - $t = \cdot(L)$ . Then  $\text{QSub}(t) = \{t\} \cup \bigcup_{p \in \text{elems}(L)} \text{QSub}(p)$ . And  $\text{QSub}(\text{QSub}(t)) = \text{QSub}(t) \cup \bigcup_{p \in \text{elems}(L)} \text{QSub}(\text{QSub}(p))$ , but since  $\text{size}_{\text{DAG}}(p) < k$  for such any  $p$  we can apply induction supposition and get  $\bigcup_{p \in \text{elems}(L)} \text{QSub}(\text{QSub}(p)) = \bigcup_{p \in \text{elems}(L)} \text{QSub}(p) = \text{QSub}(t) \setminus \{t\}$ . Then  $\text{QSub}(\text{QSub}(t)) = \text{QSub}(t) \cup (\text{QSub}(t) \setminus \{t\}) = \text{QSub}(t)$ .

**Statement 15:** By induction on  $\text{size}_{\text{DAG}}(t)$

- $\text{size}_{\text{DAG}}(t) = 1$ .
  - $t \in \mathcal{A}$ . As  $t\sigma = t$  and  $\text{Vars}(t) = \emptyset$ , the statement becomes trivial.
  - $t \in \mathcal{X}$ . Then  $\text{Sub}(t)\sigma = t\sigma$ ,  $\text{Vars}(t) = \{t\}$ ; and as for any term  $p$ ,  $p \in \text{Sub}(p)$ , we have  $\text{Sub}(t\sigma) = \{t\sigma\} \cup \text{Sub}(t\sigma)$ .
- Suppose, that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k \geq 1$ ), the statement is true.
- Given a term  $t : \text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ . Let us consider all possible cases:
  - $t = \text{bin}(t_1, t_2)$ . Then  $t\sigma = \text{bin}(t_1\sigma, t_2\sigma)$  and  $\text{Vars}(t) = \text{Vars}(t_1) \cup \text{Vars}(t_2)$ .  $\text{Sub}(t\sigma) = \{t\sigma\} \cup \text{Sub}(t_1\sigma) \cup \text{Sub}(t_2\sigma) =$  (as  $\text{size}_{\text{DAG}}(t_i) < k$ )  $= \{t\sigma\} \cup \text{Sub}(t_1)\sigma \cup \text{Sub}(\text{Vars}(t_1)\sigma) \cup$

$$\begin{aligned}
& \text{Sub}(t_2)\sigma \cup \text{Sub}(\text{Vars}(t_2)\sigma) = \{t\sigma\} \cup \text{Sub}(t_1)\sigma \cup \text{Sub}(t_2)\sigma \cup \\
& \text{Sub}((\text{Vars}(t_1) \cup \text{Vars}(t_2))\sigma) = \text{Sub}(t)\sigma \cup \text{Sub}(\text{Vars}(t)\sigma). \\
& - t = \text{priv}(t_1). \text{ Proof is similar to one for the case above.} \\
& - t = \cdot(\{t_1, \dots, t_m\}). \text{ We have } t\sigma = \cdot(\{t_1\sigma, \dots, t_m\sigma\}) \text{ and} \\
& \text{Vars}(t) = \bigcup_{i=1, \dots, m} \text{Vars}(t_i). \text{ Then we have } \text{Sub}(t\sigma) = \{t\sigma\} \cup \\
& \bigcup_{i=1, \dots, m} \text{Sub}(t_i\sigma) = (\text{as } \text{size}_{\text{DAG}}(t_i) < k) \\
& = \{t\sigma\} \cup \bigcup_{i=1, \dots, m} (\text{Sub}(t_i)\sigma \cup \text{Sub}(\text{Vars}(t_i)\sigma)) = \{t\sigma\} \cup \\
& \bigcup_{i=1, \dots, m} \text{Sub}(t_i)\sigma \cup \text{Sub}\left(\left(\bigcup_{i=1, \dots, m} \text{Vars}(t_i)\right)\sigma\right) = \\
& \text{Sub}(t)\sigma \cup \text{Sub}(\text{Vars}(t)\sigma).
\end{aligned}$$

**Statement 16:** It follows from the fact that  $f(t) = \lceil t \rceil$  and  $g_\sigma(t) = t\sigma$  are deterministic functions, and thus return at most one value for one given argument.

**Statement 17:** First we prove that  $\text{elems}(t) \subseteq \text{QSub}(t)$ . We use induction on  $\text{size}_{\text{DAG}}(t)$ .

- If  $\text{root}(t) \neq \cdot$ , then  $\text{elems}(t) = \{t\} \subseteq \text{QSub}(t)$ . This case includes all  $t$  such that  $\text{size}_{\text{DAG}}(t) = 1$ . Thus we need to consider only  $t = \cdot(L)$ .
- Suppose that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k \geq 1$ ), the statement holds.
- If for some  $t$  we have  $\text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ , then  $\text{elems}(t) = \bigcup_{p \in L} \text{elems}(p)$  and  $\text{QSub}(t) = \{t\} \cup \bigcup_{p \in L} \text{QSub}(p)$ . And since  $\text{size}_{\text{DAG}}(p) < k$  using the induction supposition we obtain the wanted statement.

Now we show that  $\text{QSub}(t) \subseteq \text{Sub}(t)$ . Again, applying proof by induction on  $\text{size}_{\text{DAG}}(t)$  we have:

- If  $\text{size}_{\text{DAG}}(t) = 1$ , then  $\text{QSub}(t) = \text{Sub}(t) = \{t\}$ .
- Suppose that for any  $t : \text{size}_{\text{DAG}}(t) < k$  ( $k \geq 1$ ), the statement holds.
- If for some  $t$  we have  $\text{size}_{\text{DAG}}(t) = k$ ,  $k > 1$ , then
  - $t = \text{bin}(t_1, t_2)$ . Then  $\text{QSub}(t) = \{t\} \cup \text{QSub}(t_1) \cup \text{QSub}(t_2)$  and  $\text{Sub}(t) = \{t\} \cup \text{Sub}(t_1) \cup \text{Sub}(t_2)$ , where  $\max\{\text{size}_{\text{DAG}}(t_1), \text{size}_{\text{DAG}}(t_2)\} < k$ . And then using induction supposition we can conclude for this case.

- $t = \text{priv}(t_1)$ . Proof is similar to one for the case above.
- $t = \cdot(\{t_1, \dots, t_m\})$ . Then we have
 
$$\begin{aligned} \text{QSub}(t) &= \{t\} \cup \bigcup_{p \in \text{elems}(\{t_1, \dots, t_m\})} \text{QSub}(p) \subseteq \text{(using the already proved part of the property)} \\ &\subseteq \{t\} \cup \bigcup_{p \in \text{QSub}(\{t_1, \dots, t_m\})} \text{QSub}(p) = \text{(as } \text{QSub}(\text{QSub}(t)) = \text{QSub}(t)) \\ &= \{t\} \cup \bigcup_{p \in \{t_1, \dots, t_m\}} \text{QSub}(p) \subseteq \text{(by induction supposition, as } \text{size}_{\text{DAG}}(t_i) < k \text{ for all } i) \\ &\subseteq \{t\} \cup \bigcup_{p \in \{t_1, \dots, t_m\}} \text{Sub}(p) = \text{Sub}(t). \end{aligned}$$

**Statement 18:** Using Statement 4 and the fact that  $\ulcorner \cdot \urcorner$  is a deterministic function we obtain  $\forall p, q \in \text{Sub}(\ulcorner t \urcorner) p \neq q \exists p', q' \in \text{Sub}(t) : p = \ulcorner p' \urcorner \wedge q = \ulcorner q' \urcorner \wedge p \neq q$ . And thus,  $|\text{Sub}(\ulcorner t \urcorner)| \leq |\text{Sub}(t)|$ .

**Statement 19:** We have  $\text{QSub}(\cdot(\{t_1, \dots, t_l\})) = \{\cdot(\{t_1, \dots, t_l\})\} \cup \bigcup_{i=1}^l \text{QSub}(\text{elems}(t_i))$ . Using Statement 17 and 12 we have  $\text{QSub}(\text{elems}(t_i)) \subseteq \text{QSub}(\text{QSub}(t_i)) = \text{QSub}(t_i)$ . Thus,  $\text{QSub}(\cdot(\{t_1, \dots, t_l\})) \subseteq \{\cdot(\{t_1, \dots, t_l\})\} \cup \text{QSub}(t_1) \cdots \cup \text{QSub}(t_l)$ .

### Appendix B.2. Proof of Property 11

As was stated before, the measure of the problem input is  $\text{measure}(\mathcal{S}) = n \times \text{size}_{\text{DAG}}(\mathcal{S}) + |\mathbb{E}(\mathcal{S})|$ , where  $\mathcal{S} = \{E_i \triangleright t_i\}_{i=1, \dots, n}$ .

Algorithm 2 returns a normalized proof  $\sigma$  for decision problem if it exists. Moreover,  $\text{size}_{\text{DAG}}(x\sigma) \leq 2 \times \text{size}_{\text{DAG}}(\mathcal{S})$  for any  $x \in \text{Vars}(\mathcal{S})$ .

First, we will normalize  $\mathcal{S}\sigma$ . From Lemma 18 follows, that we can do it for the time  $T_{\ulcorner \cdot \urcorner} \leq P_{\ulcorner \cdot \urcorner}(\text{measure}(\mathcal{S}\sigma))$ , where  $P_{\ulcorner \cdot \urcorner}$  is some polynomial with non-negative coefficients of some degree  $m'' > 0$ .

From the Proposition 12 we will know that check of derivability of a normalized ground term  $g$  from set of normalized ground terms  $G$  takes a polynomial time depending on  $\text{size}_{\text{DAG}}(G \cup \{g\})$ . That is, there exists a polynomial  $P_g$  with non-negative coefficients, such that number of operations (execution time) to verify the derivability ( $g$  from  $G$ ) will be limited by  $P_g(\text{size}_{\text{DAG}}(G \cup \{g\}))$ . Then the execution time for checking a set of ground constraints  $\{G_i \triangleright g_i\}_{i=1, \dots, n}$  will be limited by  $\sum_{i=1}^n P_g(\text{size}_{\text{DAG}}(G_i \cup \{g_i\}))$ .

To show that the algorithm is in  $NP$  we need to show, that execution time of check is polynomial limited by measure of algorithm's input, i.e. there exists a polynomial  $P$ , such that execution time does not exceed  $O(P(n \times \text{size}_{\text{DAG}}(\mathcal{S}) + |\mathbb{E}(\mathcal{S})|))$  steps.

In our case, execution time  $T$  of a check will be  $T = T_{\neg} + T_g$ , where  $T_g$  is a time needed for checking ground derivability of  $\mathcal{S}\sigma$ :  $T_g \leq \sum_{i=1}^n P_g(\text{size}_{\text{DAG}}(\neg(E_i \cup \{t_i\})\sigma))$ . As  $P_g$  is a polynomial, let us say, of degree  $m' > 0$ , with non-negative coefficients, we can use the fact, that for any positive integers  $x_1, \dots, x_k$  we have  $\sum_{i=1}^k P_g(x_i) \leq P_g(\sum_{i=1}^k x_i)$ . Then we have  $T_g \leq P_g(\sum_{i=1}^n \text{size}_{\text{DAG}}(\neg(E_i \cup \{t_i\})\sigma))$  and by Statement 18 of Lemma 4 we have  $T_g \leq P_g(\sum_{i=1}^n \text{size}_{\text{DAG}}((E_i \cup \{t_i\})\sigma))$ ; using the same lemma, we have

$$\begin{aligned}
T_g &\leq P_g \left( \sum_{i=1}^n \left( \text{size}_{\text{DAG}}(E_i \cup \{t_i\}) + \text{size}_{\text{DAG}} \left( \bigcup_x x\sigma \right) \right) \right) \leq \\
&\leq P_g \left( \sum_{i=1}^n \left( \text{size}_{\text{DAG}}(E_i) + \text{size}_{\text{DAG}}(t_i) + \sum_x \text{size}_{\text{DAG}}(x\sigma) \right) \right) \leq \\
&\leq P_g \left( \sum_{i=1}^n (2 \text{size}_{\text{DAG}}(\mathcal{S})) + n \times \sum_x (\text{size}_{\text{DAG}}(x\sigma)) \right) \leq \\
&\leq P_g \left( 2 \times n \times \text{size}_{\text{DAG}}(\mathcal{S}) + n \times \sum_x (2 \times \text{size}_{\text{DAG}}(\mathcal{S})) \right) \leq \\
&\leq P_g (2 \times n \times \text{size}_{\text{DAG}}(\mathcal{S}) + 2 \times n \times (\text{size}_{\text{DAG}}(\mathcal{S}))^2) \leq \\
&\leq P_g (4 \times n \times (\text{size}_{\text{DAG}}(\mathcal{S}))^2) \leq P_g (4 \times (n \times \text{size}_{\text{DAG}}(\mathcal{S}) + |\mathbb{E}(\mathcal{S})|)^2) = \\
&= O \left( (\text{measure}(\mathcal{S}))^{2m'} \right).
\end{aligned}$$

On the other hand, let us consider  $T_{\neg}$ .

We have  $T_{\neg} \leq P_{\neg}(n \times \text{size}_{\text{DAG}}(\mathcal{S}\sigma) + |\mathbb{E}(\mathcal{S}\sigma)|)$ . One can see that the number of edges in DAG-representation of  $\mathcal{S}\sigma$  (where every variable  $x$  of  $\mathcal{S}$  is replaced by  $x\sigma$ ) will not exceed the number of edges in  $\mathcal{S}$  plus the number of edges of all  $x\sigma$ :  $|\mathbb{E}(\mathcal{S}\sigma)| \leq |\mathbb{E}(\mathcal{S})| + \sum_{x \in \text{Vars}(\mathcal{S})} |\mathbb{E}(x\sigma)|$ . And since  $\sigma$  is normalized, we can use Lemma 17:  $T_{\neg} \leq P_{\neg}(n \times \text{size}_{\text{DAG}}(\mathcal{S}\sigma) + |\mathbb{E}(\mathcal{S})| + \sum_{x \in \text{Vars}(\mathcal{S})} (\text{size}_{\text{DAG}}(x\sigma))^2)$ .

Then, using Lemma 4 (Statement 15) we obtain  $\text{Sub}(\mathcal{S}\sigma) = \text{Sub}(\mathcal{S})\sigma \cup \text{Sub}(\text{Vars}(\mathcal{S})\sigma)$ , and thus,  $\text{size}_{\text{DAG}}(\mathcal{S}\sigma) \leq |\text{Sub}(\mathcal{S})\sigma| + \sum_{x \in \text{Vars}(\mathcal{S})} \text{size}_{\text{DAG}}(x\sigma)$ . From Statement 16 of Lemma 4 follows that  $|\text{Sub}(\mathcal{S})\sigma| \leq \text{size}_{\text{DAG}}(\mathcal{S})$ . Since  $\text{size}_{\text{DAG}}(x\sigma) \leq 2 \times \text{size}_{\text{DAG}}(\mathcal{S})$  and  $|\text{Vars}(\mathcal{S})| \leq \text{size}_{\text{DAG}}(\mathcal{S})$ , we obtain  $\text{size}_{\text{DAG}}(\mathcal{S}\sigma) \leq \text{size}_{\text{DAG}}(\mathcal{S}) + 2 \times (\text{size}_{\text{DAG}}(\mathcal{S}))^2$ . In the same way,  $\sum_{x \in \text{Vars}(\mathcal{S})} (\text{size}_{\text{DAG}}(x\sigma))^2 \leq \text{size}_{\text{DAG}}(\mathcal{S}) \times (2 \times \text{size}_{\text{DAG}}(\mathcal{S}))^2$ .

Therefore,  $T_{\neg} \leq P_{\neg}(n \times (\text{size}_{\text{DAG}}(\mathcal{S}) + 2 \times (\text{size}_{\text{DAG}}(\mathcal{S}))^2) + |\mathbb{E}(\mathcal{S})| + \text{size}_{\text{DAG}}(\mathcal{S}) \times (2 \times \text{size}_{\text{DAG}}(\mathcal{S}))^2) = O((\text{measure}(\mathcal{S}))^{3m''})$ .

Thus,  $T = O((\text{measure}(\mathcal{S}))^{3m''} + (\text{measure}(\mathcal{S}))^{2m'})$  that shows, that a test of a proof returned by the algorithm takes polynomial time what gives us a class of complexity.

### Appendix B.3. Proof of Lemma 12

Let us consider all the cases of DY+ACI rules:

- $t_1, t_2 \rightarrow \neg \text{pair}(t_1, t_2) \neg$  We have two cases:
  - $\exists u \in \text{QSub}(S)$  such that  $\neg \text{pair}(t_1, t_2) \neg = \neg u \sigma \neg$ . Then we have  $\pi(H(\neg \text{pair}(t_1, t_2) \neg)) = \pi(H(\text{pair}(\neg t_1 \neg, \neg t_2 \neg))) = \pi(\{\text{pair}(\pi(H(\neg t_1 \neg)), \pi(H(\neg t_2 \neg)))\}) = \neg \text{pair}(\pi(H(t_1)), \pi(H(t_2))) \neg$  and then  $\pi(H(\neg \text{pair}(t_1, t_2) \neg)) \in \text{Der}(\{\pi(H(t_1)), \pi(H(t_2))\})$ .
  - $\nexists u \in \text{QSub}(S)$  such that  $\neg \text{pair}(t_1, t_2) \neg = \neg u \sigma \neg$ . Then (by definition, Lemma 4 and Proposition 1)  $\pi(H(\neg \text{pair}(t_1, t_2) \neg)) = \pi(H(\neg t_1 \neg \cup H(\neg t_2 \neg))) \in \text{Der}(\neg H(t_1) \cup H(t_2) \neg)$ . By Proposition 1,  $\neg H(t_1) \neg \subseteq \text{Der}(\{\pi(H(t_1))\})$  and  $\neg H(t_2) \neg \subseteq \text{Der}(\{\pi(H(t_2))\})$ , then by Lemma 3,  $\neg H(t_1) \neg \cup \neg H(t_2) \neg \subseteq \text{Der}(\{\pi(H(t_1))\} \cup \{\pi(H(t_2))\})$ . Now, by applying Lemma 2, we have  $\pi(H(\neg \text{pair}(t_1, t_2) \neg)) \in \text{Der}(\{\pi(H(t_1))\} \cup \{\pi(H(t_2))\})$ .

So, in this case  $\pi(H(r)) \in \text{Der}(\{\pi(H(l_1)), \pi(H(l_2))\})$ .

- $t_1, t_2 \rightarrow \neg \text{enc}(t_1, t_2) \neg$ . Proof of this case can be done by analogy of previous one.
- $\{t_1, t_2\} \rightarrow \neg \text{aenc}(t_1, t_2) \neg$ . The same.
- $t_1, \text{priv}(t_2) \rightarrow \neg \text{sig}(t_1, \text{priv}(t_2)) \neg$ .
  - $\exists u \in \text{QSub}(S)$  such that  $\neg \text{sig}(t_1, \text{priv}(t_2)) \neg = \neg u \sigma \neg$ . Then  $\pi(H(\neg \text{sig}(t_1, \text{priv}(t_2)) \neg)) = \pi(H(\text{sig}(t_1, \text{priv}(t_2)))) = \pi(\{\text{sig}(\pi(H(\neg t_1 \neg)), \pi(H(\neg \text{priv}(t_2) \neg)))\}) = \neg \text{sig}(\pi(H(t_1)), \text{priv}(\pi(H(t_2)))) \neg$  and then  $\pi(H(\neg \text{sig}(t_1, \text{priv}(t_2)) \neg)) \in \text{Der}(\{\pi(H(t_1)), \pi(H(\text{priv}(t_2)))\})$  (as  $\pi(H(\text{priv}(t_2))) = \text{priv}(\pi(H(t_2)))$ ).

- $\nexists u \in \mathsf{QSub}(S)$  such that  $\ulcorner \text{sig}(t_1, \text{priv}(t_2)) \urcorner = \ulcorner u\sigma \urcorner$ . This case can be proved in similar way as done for  $\{t_1, t_2\} \rightarrow \ulcorner \text{pair}(t_1, t_2) \urcorner$ .
- $t_1, \dots, t_m \rightarrow \ulcorner \cdot(t_1, \dots, t_m) \urcorner$ . On one hand,  $\pi(H(\ulcorner \cdot(t_1, \dots, t_m) \urcorner)) = \pi(H(\cdot(t_1, \dots, t_m))) = \pi(H(t_1) \cup \dots \cup H(t_m)) \in \text{Der}(\ulcorner H(t_1) \cup \dots \cup H(t_m) \urcorner)$ . On the other hand,  $\ulcorner H(t_i) \urcorner \subseteq \text{Der}(\{\pi(H(t_i))\})$ . And thus, by Lemma 3,  $\pi(H(\ulcorner \cdot(t_1, \dots, t_m) \urcorner)) \in \text{Der}(\{\pi(H(t_1)), \dots, \pi(H(t_m))\})$ .
- $\text{enc}(t_1, t_2), \ulcorner t_2 \urcorner \rightarrow \ulcorner t_1 \urcorner$ . Here we have to show that  $\pi(H(\ulcorner t_1 \urcorner))$  is derivable from  $\{\pi(H(\text{enc}(t_1, t_2))), \pi(H(\ulcorner t_2 \urcorner))\}$ . Consider two cases:
  - $\exists u \in \mathsf{QSub}(S)$  such that  $\ulcorner \text{enc}(t_1, t_2) \urcorner = \ulcorner u\sigma \urcorner$ . Then  $\pi(H(\text{enc}(t_1, t_2))) = \text{enc}(\pi(H(t_1)), \pi(H(t_2)))$ , and  $\pi(H(\ulcorner t_1 \urcorner)) = \pi(H(t_1)) \in \text{Der}(\{\text{enc}(\pi(H(t_1)), \pi(H(t_2))), \ulcorner \pi(H(\ulcorner t_2 \urcorner)) \urcorner\})$ .
  - $\nexists u \in \mathsf{QSub}(S)$  such that  $\ulcorner \text{enc}(t_1, t_2) \urcorner = \ulcorner u\sigma \urcorner$ . Then  $\pi(H(\text{enc}(t_1, t_2))) = \pi(H(t_1) \cup H(t_2))$ . Using Proposition 1, we have  $\ulcorner H(t_1) \cup H(t_2) \urcorner \subseteq \text{Der}(\{\pi(H(\text{enc}(t_1, t_2)))\})$ , thus (by Lemma 4)  $\ulcorner H(t_1) \urcorner \subseteq \text{Der}(\{\pi(H(\text{enc}(t_1, t_2)))\})$ . And then, by Proposition 1 we have that  $\pi(H(t_1)) \in \text{Der}(\ulcorner H(t_1) \urcorner)$ . Therefore, by Lemma 2, we have  $\pi(H(\ulcorner t_1 \urcorner)) = \pi(H(t_1)) \in \text{Der}(\pi(H(\text{enc}(t_1, t_2))))$ .
- $\text{aenc}(t_1, t_2), \ulcorner \text{priv}(t_2) \urcorner \rightarrow \ulcorner t_1 \urcorner$ . Here we have to show that  $\pi(H(\ulcorner t_1 \urcorner))$  is derivable from  $\{\pi(H(\text{aenc}(t_1, t_2))), \pi(H(\ulcorner \text{priv}(t_2) \urcorner))\}$ . Consider two cases:
  - $\exists u \in \mathsf{QSub}(S)$  such that  $\ulcorner \text{aenc}(t_1, t_2) \urcorner = \ulcorner u\sigma \urcorner$ . Then  $\pi(H(\text{aenc}(t_1, t_2))) = \text{aenc}(\pi(H(t_1)), \pi(H(t_2)))$ , and then  $\pi(H(\ulcorner t_1 \urcorner)) = \pi(H(t_1)) \in \text{Der}(\{\text{aenc}(\pi(H(t_1)), \pi(H(t_2))), \ulcorner \text{priv}(\pi(H(t_2))) \urcorner\})$ . On the other hand,  $\pi(H(\ulcorner \text{priv}(t_2) \urcorner)) = \pi(H(\text{priv}(t_2))) = \pi(\{\text{priv}(\pi(H(t_2)))\}) = \ulcorner \text{priv}(\pi(H(t_2))) \urcorner$ .
  - $\nexists u \in \mathsf{QSub}(S)$  such that  $\ulcorner \text{aenc}(t_1, t_2) \urcorner = \ulcorner u\sigma \urcorner$ . Then  $\pi(H(\text{aenc}(t_1, t_2))) = \pi(H(t_1) \cup H(t_2))$ . Using Proposition 1, we have  $\ulcorner H(t_1) \cup H(t_2) \urcorner \subseteq \text{Der}(\{\pi(H(\text{aenc}(t_1, t_2)))\})$ , thus (by Lemma 4)  $\ulcorner H(t_1) \urcorner \subseteq \text{Der}(\{\pi(H(\text{aenc}(t_1, t_2)))\})$ . And then, by Proposition 1 we have that  $\pi(H(t_1)) \in \text{Der}(\ulcorner H(t_1) \urcorner)$ . Therefore, by Lemma 2,  $\pi(H(\ulcorner t_1 \urcorner)) = \pi(H(t_1)) \in \text{Der}(\pi(H(\text{aenc}(t_1, t_2))))$ .

- $\text{pair}(t_1, t_2) \rightarrow \ulcorner t_1 \urcorner$ . Here, as usual, we consider two cases:
  - $\exists u \in \text{QSub}(S)$  such that  $\ulcorner \text{pair}(t_1, t_2) \urcorner = \ulcorner u\sigma \urcorner$ . Then  $\pi(H(\text{pair}(t_1, t_2))) = \text{pair}(\pi(H(t_1)), \pi(H(t_2)))$  and then  $\pi(H(\ulcorner t_1 \urcorner)) = \ulcorner \pi(H(t_1)) \urcorner \in \text{Der}(\{\pi(H(\text{pair}(t_1, t_2)))\})$ .
  - $\nexists u \in \text{QSub}(S)$  such that  $\ulcorner \text{pair}(t_1, t_2) \urcorner = \ulcorner u\sigma \urcorner$ . Then  $\pi(H(\text{pair}(t_1, t_2))) = \pi(H(t_1) \cup H(t_2))$ . Then by Proposition 1, we have  $\ulcorner H(t_1) \cup H(t_2) \urcorner \subseteq \text{Der}(\{\pi(H(\text{pair}(t_1, t_2)))\})$ , thus  $\ulcorner H(t_1) \urcorner \subseteq \text{Der}(\{\pi(H(\text{pair}(t_1, t_2)))\})$ . And then, by Proposition 1 we have that  $\pi(H(t_1)) \in \text{Der}(\ulcorner H(t_1) \urcorner)$ . Therefore, by Lemma 2,  $\pi(H(\ulcorner t_1 \urcorner)) = \pi(H(t_1)) \in \text{Der}(\pi(H(\text{pair}(t_1, t_2))))$ .
- $\text{pair}(t_1, t_2) \rightarrow \ulcorner t_2 \urcorner$ . Proof like above.
- $\cdot(t_1, \dots, t_m) \rightarrow \ulcorner t_i \urcorner$ . We have  $\pi(H(\cdot(t_1, \dots, t_m))) = \pi(H(t_1) \cup \dots \cup H(t_m))$ . Then by Proposition 1,  $\ulcorner H(t_1) \cup \dots \cup H(t_m) \urcorner \subseteq \text{Der}(\pi(H(\cdot(t_1, \dots, t_m))))$ ; thus  $\ulcorner H(t_i) \urcorner \subseteq \text{Der}(\pi(H(\cdot(t_1, \dots, t_m))))$ . As  $\pi(H(t_i)) \in \text{Der}(\ulcorner H(t_i) \urcorner)$ , by Lemma 2 we have  $\pi(H(\ulcorner t_i \urcorner)) = \pi(H(t_i)) \in \text{Der}(\pi(H(\cdot(t_1, \dots, t_m))))$ .

As all possible cases satisfy lemma conditions, we proved the lemma.

#### Appendix B.4. Proof of Property 8

*Proof.* From proposition 2 and 3 we know that if  $\sigma'$  is a model of  $\mathcal{S}$  then  $\ulcorner \sigma' \urcorner$  is a model of  $\mathcal{S}$  and  $\ulcorner \sigma' \urcorner$  is a model of  $\ulcorner \mathcal{S} \urcorner$ . Then, there exists a substitution  $\theta : \text{dom}(\theta) = \text{dom}(\ulcorner \sigma' \urcorner)$ ,  $\text{dom}(\theta) \subseteq \text{dom}(\theta)$ ,  $\sigma'' = \ulcorner \sigma' \urcorner|_{\text{dom}(\theta)\theta}$  and  $\sigma''$  is a model of  $\ulcorner \mathcal{S} \urcorner \theta$  such that  $x\sigma'' \neq y\sigma''$ , if  $x \neq y$  (this is true because we can show how to build  $\theta$  : given the  $\ulcorner \sigma' \urcorner$  — simply split  $\text{dom}(\ulcorner \sigma' \urcorner)$  into the classes of equivalence modulo  $\ulcorner \sigma' \urcorner$ , i.e.  $x \equiv y \iff x\ulcorner \sigma' \urcorner = y\ulcorner \sigma' \urcorner$ ; for every class choose one representative  $[x]_{\equiv}$ , and then  $x\theta = [x]_{\equiv}$ ). Note, that  $\theta\sigma'' = \sigma'$ , that's why  $\sigma''$  is a model of  $\ulcorner \mathcal{S} \urcorner \theta$ .

Then, as  $\sigma''$  is a model of  $\ulcorner \mathcal{S} \urcorner \theta$ , using Proposition 2, we can say that  $\sigma''$  is a model of  $\ulcorner \ulcorner \mathcal{S} \urcorner \theta \urcorner$ . Moreover,  $\sigma''$  is normalized and  $x\sigma'' \neq y\sigma''$  for all  $x, y \in \text{dom}(\sigma'')$  such that  $x \neq y$ . Then, we can apply Corollary 1, which gives us existence of conservative model  $\delta$  of  $\ulcorner \ulcorner \mathcal{S} \urcorner \theta \urcorner$ . That is why we can apply Proposition 7: for any  $x \in \text{Vars}(\ulcorner \ulcorner \mathcal{S} \urcorner \theta \urcorner)$ ,  $\text{size}_{\text{DAG}}(x\delta) \leq 2 \times \text{size}_{\text{DAG}}(\ulcorner \ulcorner \mathcal{S} \urcorner \theta \urcorner)$ .

Note, that using Proposition 2, Lemma 16 and definition of “model”, we can easily show that  $\delta[\theta]$  is a model of  $\ulcorner \mathcal{S} \urcorner$ . Moreover,  $\delta[\theta]$  is normalized.



By definition of  $\delta[\theta]$  we can say, that for all  $x \in \text{dom}(\delta[\theta])$  there exists  $y \in \text{dom}(\theta)$  such that  $x\delta[\theta] = y\delta$ ; and as  $y \in \mathcal{X}$  (by definition of  $\theta$ ), then  $\text{size}_{\text{DAG}}(x\delta[\theta]) = \text{size}_{\text{DAG}}(y\delta) \leq 2 \times \text{size}_{\text{DAG}}(\ulcorner \ulcorner \mathcal{S}^\top \theta^\top \urcorner \urcorner) \leq 2 \times \text{size}_{\text{DAG}}(\ulcorner \mathcal{S}^\top \urcorner)$ . Applying Lemma 15, we have  $\text{size}_{\text{DAG}}(x\delta[\theta]) \leq 2 \times \text{size}_{\text{DAG}}(\ulcorner \mathcal{S}^\top \urcorner)$ .

Summing up, we have a normalized model  $\sigma = \delta[\theta]$  of  $\ulcorner \mathcal{S}^\top \urcorner$  such that for all  $x \in \text{dom}(\sigma)$ ,  $\text{size}_{\text{DAG}}(x\sigma) \leq 2 \times \text{size}_{\text{DAG}}(\ulcorner \mathcal{S}^\top \urcorner)$ . □

## References

- [1] J. Millen, V. Shmatikov, [Constraint solving for bounded-process cryptographic protocol analysis](#), in: Proceedings of the 8th ACM conference on Computer and Communications Security, CCS '01, ACM, New York, NY, USA, 2001, pp. 166–175. [doi:http://doi.acm.org/10.1145/501983.502007](http://doi.acm.org/10.1145/501983.502007).  
URL <http://doi.acm.org/10.1145/501983.502007>
- [2] D. Basin, S. Mödersheim, L. Viganò, Ofmc: A symbolic model checker for security protocols, *International Journal of Information Security* 4 (2005) 181–208.
- [3] M. Turuani, The CL-Atse Protocol Analyser, in: *Term Rewriting and Applications (RTA)*, 2006, pp. 277–286.
- [4] C. Cremers, The Scyther Tool: Verification, falsification, and analysis of security protocols, in: *Computer Aided Verification, 20th International Conference, CAV 2008, Princeton, USA, Proc., Vol. 5123/2008 of Lecture Notes in Computer Science*, Springer, 2008, pp. 414–418. [doi:10.1007/978-3-540-70545-1\\_38](http://doi.org/10.1007/978-3-540-70545-1_38).
- [5] D. Basin, S. Mödersheim, L. Viganò, Algebraic intruder deductions, in: *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, 2005, pp. 549–564.
- [6] Y. Chevalier, M. Rusinowitch, [Symbolic protocol analysis in the union of disjoint intruder theories: Combining decision procedures](#), *Theoretical Computer Science* 411 (10) (2010) 1261 – 1282, *iCALP 2005 - Track C: Security and Cryptography Foundations*. [doi:DOI:10.1016/j.tcs.2009.10.022](http://doi.org/10.1016/j.tcs.2009.10.022).

- URL <http://www.sciencedirect.com/science/article/B6V1G-4XKBYW5-2/2/1191e694e6322c89670136e608d98620>
- [7] V. Cortier, S. Delaune, P. Lafourcade, [A survey of algebraic properties used in cryptographic protocols](#), Journal of Computer Security 14 (1) (2006) 1–43.  
URL <http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/surveyCDL.pdf>
  - [8] Y. Chevalier, R. Küsters, M. Rusinowitch, M. Turuani, An np decision procedure for protocol insecurity with xor, Theor. Comput. Sci. 338 (1-3) (2005) 247–274.
  - [9] S. Delaune, P. Lafourcade, D. Lugiez, R. Treinen, [Symbolic protocol analysis for monoidal equational theories](#), Information and Computation 206 (2-4) (2008) 312–351. doi:10.1016/j.ic.2007.07.005.  
URL <http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/DLLT-ic07.pdf>
  - [10] L. Mazaré, Satisfiability of Dolev-Yao Constraints, Electronic Notes in Theoretical Computer Science 125 (1) (2005) 109–124.
  - [11] L. Mazaré, [Computational Soundness of Symbolic Models for Cryptographic Protocols](#), Ph.D. thesis, Institut National Polytechnique de Grenoble (October 2006).  
URL <http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/these-mazare.pdf>
  - [12] P. Syverson, C. Meadows, I. Cervesato, Dolev-Yao is no better than Machiavelli, in: First Workshop on Issues in the Theory of Security — WITS’00, 2000, pp. 87–92.
  - [13] M. Baudet, Deciding security of protocols against off-line guessing attacks, in: ACM Conference on Computer and Communications Security, 2005, pp. 16–25.
  - [14] M. Rusinowitch, M. Turuani, Protocol insecurity with a finite number of sessions, composed keys is np-complete, Theor. Comput. Sci. 1-3 (299) (2003) 451–475.

- [15] Y. Chevalier, R. Küsters, M. Rusinowitch, M. Turuani, Deciding the security of protocols with diffie-hellman exponentiation and products in exponents, in: FSTTCS, 2003, pp. 124–135.
- [16] H. Comon-Lundh, V. Shmatikov, Intruder deductions, constraint solving and insecurity decision in presence of exclusive or, IEEE Comp. Soc. Press, 2003, pp. 271–280.
- [17] V. Shmatikov, Decidable analysis of cryptographic protocols with products and modular exponentiation, in: In Proc. 13th European Symposium on Programming (ESOP '04), volume 2986 of LNCS, Springer-Verlag, 2004, pp. 355–369.
- [18] S. Delaune, [Vérification des protocoles cryptographiques et propriétés algébriques](#), Thèse de doctorat, Laboratoire Spécification et Vérification, ENS Cachan, France (Jun. 2006).  
URL <http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/these-delaune.pdf>
- [19] S. Bursuc, H. Comon-lundh, S. Delaune, Associative-commutative deducibility constraints, in: Proceedings of the 24th Annual Symposium on Theoretical Aspects of Computer Science (STACS'07), volume 4393 of Lecture Notes in Computer Science, Springer, 2007, pp. 634–645.
- [20] Y. Chevalier, D. Lugiez, M. Rusinowitch, Towards an automatic analysis of web service security, in: FroCoS 2007, Liverpool, UK, September 10-12, Vol. 4720 of Lecture Notes in Computer Science, Springer, 2007, pp. 133–147.
- [21] O. Foundation, [OWASP-DV-008, OWASP Testing Guide, v3.0](#), [http://www.owasp.org/index.php/Testing\\_for\\_XML\\_Injection\\_\(OWASP-DV-008\)](http://www.owasp.org/index.php/Testing_for_XML_Injection_(OWASP-DV-008)) (2008).  
URL [http://www.owasp.org/index.php/Testing\\_for\\_XML\\_Injection\\_\(OWASP-DV-008\)](http://www.owasp.org/index.php/Testing_for_XML_Injection_(OWASP-DV-008))